

GENERATING WITTY COMMENTS...

LET'S GO!



GETTING TO KNOW THE **AMAZING** AMAZON WEB SERVICES

Luigi Libero Lucio Starace

`luigi.starace@gmail.com`

June 4, 2018

University of Naples, Federico II

1 A little bit of context

- 1 A little bit of context
- 2 An AWS bestiarium

- 1 A little bit of context
- 2 An AWS bestiary
- 3 Serverless architectures

- 1 A little bit of context
- 2 An AWS bestiary
- 3 Serverless architectures
- 4 Demo: a serverless web application

- 1 A little bit of context
- 2 An AWS bestiary
- 3 Serverless architectures
- 4 Demo: a serverless web application
- 5 Take Home Messages

A LITTLE BIT OF CONTEXT



Cloud computing is the on-demand delivery of computing resources through a cloud services platform via the internet with pay-as-you-go pricing.

Cloud computing is the **on-demand delivery** of computing resources through a cloud services platform via the internet with pay-as-you-go pricing.

Cloud computing is the **on-demand delivery** of computing resources through a cloud services platform via the internet with **pay-as-you-go** pricing.

- Software as a Service (SaaS)

- **Software as a Service (SaaS)**

The service vendor provides the user with a completed product that is run and managed by the service provider.

- **Software as a Service (SaaS)**

The service vendor provides the user with a completed product that is run and managed by the service provider.

- **Platform as a Service (PaaS)**

- **Software as a Service (SaaS)**

The service vendor provides the user with a completed product that is run and managed by the service provider.

- **Platform as a Service (PaaS)**

The service vendor provides the user with a set of API which can be used to build, test and deploy applications.

- **Software as a Service (SaaS)**

The service vendor provides the user with a completed product that is run and managed by the service provider.

- **Platform as a Service (PaaS)**

The service vendor provides the user with a set of API which can be used to build, test and deploy applications.

- **Infrastructure as a Service (IaaS)**

- **Software as a Service (SaaS)**

The service vendor provides the user with a completed product that is run and managed by the service provider.

- **Platform as a Service (PaaS)**

The service vendor provides the user with a set of API which can be used to build, test and deploy applications.

- **Infrastructure as a Service (IaaS)**

The service vendor provides users access to computing resources such as servers, storage and networking.

SERVICE MODELS: A VISUAL COMPARISON

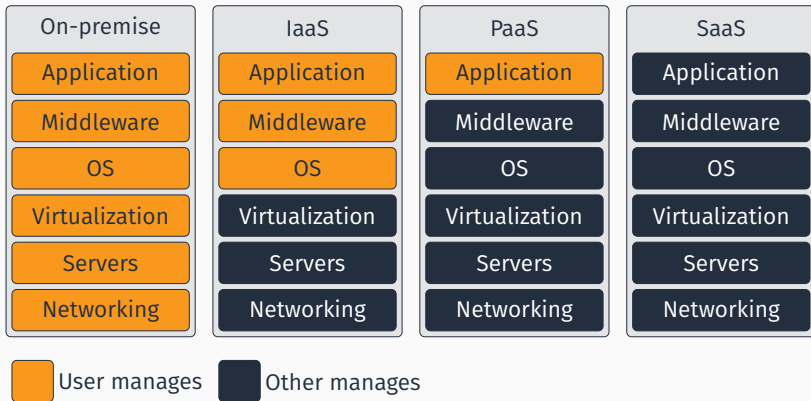
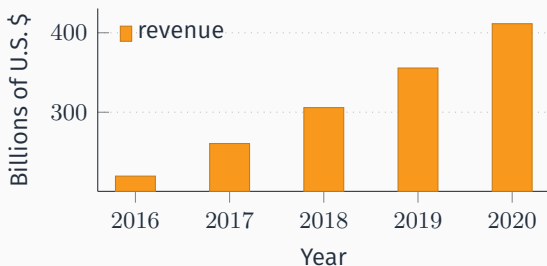


Figure 1: A service models comparison

SOME STATS

Worldwide Public Cloud Services Revenue Forecast (Billions of U.S. Dollars) [Gar17]

| 2016 | 2017 | 2018 | 2019 | 2020 |
|-------|-------|-------|-------|-------|
| 219,6 | 260,6 | 305,8 | 355,6 | 411,4 |



- Google



Google Cloud

- Google
- IBM



- Google
- IBM
- Microsoft



- Google
- IBM
- Microsoft
- Alibaba



- Google
- IBM
- Microsoft
- Alibaba
- Oracle

ORACLE[®]

CLOUD

- Google
- IBM
- Microsoft
- Alibaba
- Oracle
- Amazon



MARKET SHARE

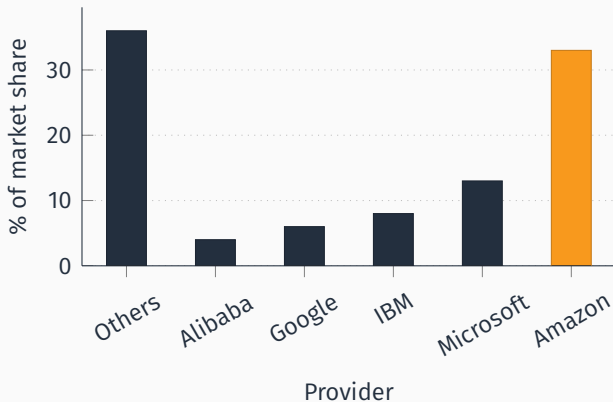


Figure 2: Market share in Q4 2017 (IaaS, PaaS, Hosted Private Cloud)
[Syn18]

AN AWS BESTIARIUM



Amazon Web Services is a collection of cloud-based services.



Amazon Web Services is a collection of cloud-based services.
A very big one.



Amazon Web Services is a collection of cloud-based services.
A VERY big one.

AN AWS BESTIARIUM



DATABASE SERVICES

RELATIONAL DATABASE SERVICE (RDS)

- Set up, operate a relational database in the cloud.



RELATIONAL DATABASE SERVICE (RDS)

- Set up, operate a relational database in the cloud.
- Takes care of backups, patching.



RELATIONAL DATABASE SERVICE (RDS)

- Set up, operate a relational database in the cloud.
- Takes care of backups, patching.
- Supports:



RELATIONAL DATABASE SERVICE (RDS)

- Set up, operate a relational database in the cloud.
- Takes care of backups, patching.
- Supports:
 - MySQL, PostgreSQL, MariaDB



RELATIONAL DATABASE SERVICE (RDS)

- Set up, operate a relational database in the cloud.
- Takes care of backups, patching.
- Supports:
 - MySQL, PostgreSQL, MariaDB
 - Oracle, MS SQL Server



RELATIONAL DATABASE SERVICE (RDS)

- Set up, operate a relational database in the cloud.
- Takes care of backups, patching.
- Supports:
 - MySQL, PostgreSQL, MariaDB
 - Oracle, MS SQL Server
 - Amazon Aurora



- DynamoDB
 - *Fast and flexible NoSQL database service for any scale.*



NON RELATIONAL DATABASE SERVICES

- DynamoDB
 - *Fast and flexible NoSQL database service for any scale.*
- ElastiCache
 - In memory data store.
 - Supports memcached, Redis



NON RELATIONAL DATABASE SERVICES

- **DynamoDB**
 - *Fast and flexible NoSQL database service for any scale.*
- **ElastiCache**
 - In memory data store.
 - Supports memcached, Redis
- **Neptune**
 - Graph database service
 - Supports RDF, SPARQL, ...



AN AWS BESTIARIUM



CLOUD STORAGE

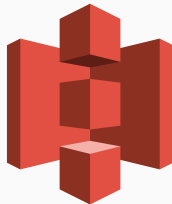
- Elastic Block Storage (EBS)
 - Persistent local storage for EC2 instances.



- Elastic Block Storage (EBS)
 - Persistent local storage for EC2 instances.
- Elastic File System (EFS)
 - File system interface to share data between EC2 instances.



- Elastic Block Storage (EBS)
 - Persistent local storage for EC2 instances.
- Elastic File System (EFS)
 - File system interface to share data between EC2 instances.
- Simple Storage Service (S3)

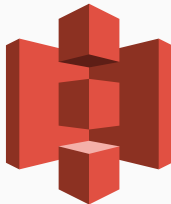


- Elastic Block Storage (EBS)
 - Persistent local storage for EC2 instances.
- Elastic File System (EFS)
 - File system interface to share data between EC2 instances.
- Simple Storage Service (S3)
- Glacier
 - Durable and cheap long-term storage.



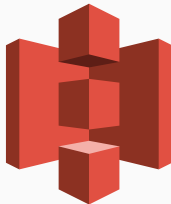
AMAZON SIMPLE STORAGE SERVICE (S3)

- *store and retrieve any amount of data from anywhere*



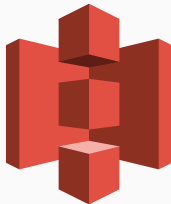
AMAZON SIMPLE STORAGE SERVICE (S3)

- *store and retrieve any amount of data from anywhere*
- 99.999999999% durability (nine nines!)



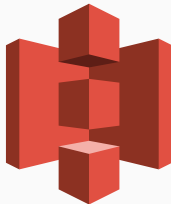
AMAZON SIMPLE STORAGE SERVICE (S3)

- *store and retrieve any amount of data from anywhere*
- 99.999999999% durability (nine nines!)
- Data is distributed across a *minimum* of three availability zones



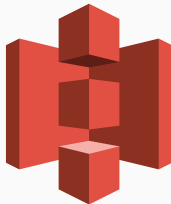
AMAZON SIMPLE STORAGE SERVICE (S3)

- *store and retrieve any amount of data from anywhere*
- 99.999999999% durability (nine nines!)
- Data is distributed across a *minimum* of three availability zones
- A logical unit of storage is a *bucket*



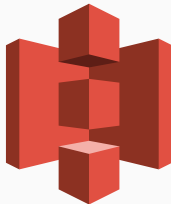
AMAZON SIMPLE STORAGE SERVICE (S3)

- *store and retrieve any amount of data from anywhere*
- 99.999999999% durability (nine nines!)
- Data is distributed across a *minimum* of three availability zones
- A logical unit of storage is a *bucket*
- Multiple storage classes



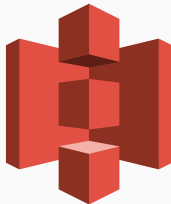
AMAZON SIMPLE STORAGE SERVICE (S3)

- *store and retrieve any amount of data from anywhere*
- 99.999999999% durability (nine nines!)
- Data is distributed across a *minimum* of three availability zones
- A logical unit of storage is a *bucket*
- Multiple storage classes
 - Standard



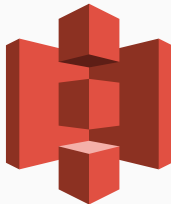
AMAZON SIMPLE STORAGE SERVICE (S3)

- *store and retrieve any amount of data from anywhere*
- 99.999999999% durability (nine nines!)
- Data is distributed across a *minimum* of three availability zones
- A logical unit of storage is a *bucket*
- Multiple storage classes
 - Standard
 - Infrequent Access



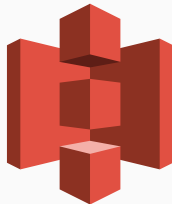
AMAZON SIMPLE STORAGE SERVICE (S3)

- *store and retrieve any amount of data from anywhere*
- 99.999999999% durability (nine nines!)
- Data is distributed across a *minimum* of three availability zones
- A logical unit of storage is a *bucket*
- Multiple storage classes
 - Standard
 - Infrequent Access
 - One zone-Infrequent Access



AMAZON SIMPLE STORAGE SERVICE (S3)

- *store and retrieve any amount of data from anywhere*
- 99.999999999% durability (nine nines!)
- Data is distributed across a *minimum* of three availability zones
- A logical unit of storage is a *bucket*
- Multiple storage classes
 - Standard
 - Infrequent Access
 - One zone-Infrequent Access
 - Amazon Glacier

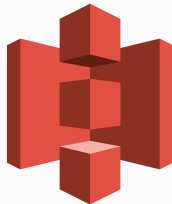


Multiple storage classes

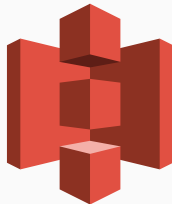
| Storage class | Storage | Retrieval (per 1K req.) |
|-------------------|-----------------|-------------------------|
| Standard | \$0.022 per GB | \$0.0004 |
| Infrequent access | \$0.0125 per GB | \$0.001 |
| IA single zone | \$0.01 per GB | \$0.001 |

Table 1: S3 pricing (Ireland)

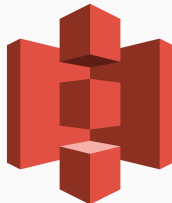
- Well-integrated with other services



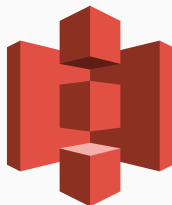
- Well-integrated with other services
 - Machine Learning



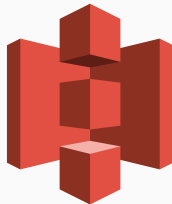
- Well-integrated with other services
 - Machine Learning
 - Big Data Analysis



- Well-integrated with other services
 - Machine Learning
 - Big Data Analysis
- REST API



- Well-integrated with other services
 - Machine Learning
 - Big Data Analysis
- REST API
- Can be used to host static websites



AN AWS BESTIARIUM



DEVELOPER TOOLS

- CodeCommit



- CodeCommit
 - Managed, scalable, private git server



- CodeCommit
 - Managed, scalable, private git server
 - Pricing based on active users (5 free, 1\$ for each additional user)



- CodeCommit
 - Managed, scalable, private git server
 - Pricing based on active users (5 free, 1\$ for each additional user)
- CodeBuild



- CodeCommit
 - Managed, scalable, private git server
 - Pricing based on active users (5 free, 1\$ for each additional user)
- CodeBuild
 - Managed, scalable build server



- CodeCommit
 - Managed, scalable, private git server
 - Pricing based on active users (5 free, 1\$ for each additional user)
- CodeBuild
 - Managed, scalable build server
 - Pay-per-minute spent building your code



- CodeDeploy



- CodeDeploy
 - Automates deployment to computing services (also to instances running on-premise)



- CodeDeploy
 - Automates deployment to computing services (also to instances running on-premise)
 - Tries to avoid downtime



- CodeDeploy
 - Automates deployment to computing services (also to instances running on-premise)
 - Tries to avoid downtime
 - 0.02\$ per-on-premise deployment



- CodeDeploy
 - Automates deployment to computing services (also to instances running on-premise)
 - Tries to avoid downtime
 - 0.02\$ per-on-premise deployment
- CodePipeline



- CodeDeploy
 - Automates deployment to computing services (also to instances running on-premise)
 - Tries to avoid downtime
 - 0.02\$ per-on-premise deployment
- CodePipeline
 - Continuous integration e continuous delivery



- CodeDeploy
 - Automates deployment to computing services (also to instances running on-premise)
 - Tries to avoid downtime
 - 0.02\$ per-on-premise deployment
- CodePipeline
 - Continuous integration e continuous delivery
 - Define your own workflow and stages

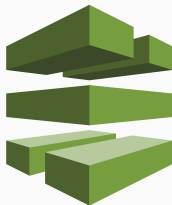


■ CodeDeploy

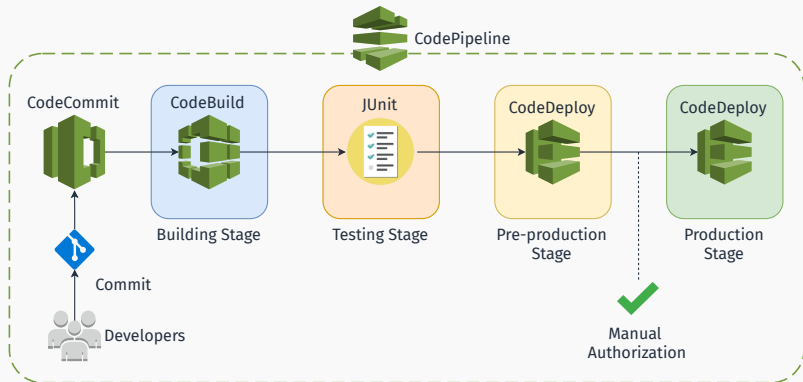
- Automates deployment to computing services (also to instances running on-premise)
- Tries to avoid downtime
- 0.02\$ per-on-premise deployment

■ CodePipeline

- Continuous integration e continuous delivery
- Define your own workflow and stages
- 1\$ per-month per active pipeline



CODEPIPELINE



Interested in CI/CD on AWS? Check these out:

- *Practicing Continuous Integration and Continuous Delivery on AWS* (whitepaper) [Ama17]
- *Set up a Continuous Deployment Pipeline using AWS CodePipeline* [Amab]
- *Tutorial: Create a Four-Stage Pipeline* [Amac]

- CodeStar



- CodeStar
 - Wrapper around developer tools to simplify setup



- CodeStar
 - Wrapper around developer tools to simplify setup
 - Templates



- CodeStar
 - Wrapper around developer tools to simplify setup
 - Templates
 - Team Management



- CodeStar
 - Wrapper around developer tools to simplify setup
 - Templates
 - Team Management
 - Central Project Dashboard



- CodeStar
 - Wrapper around developer tools to simplify setup
 - Templates
 - Team Management
 - Central Project Dashboard
 - Free of charge



- CodeStar
 - Wrapper around developer tools to simplify setup
 - Templates
 - Team Management
 - Central Project Dashboard
 - Free of charge
- Cloud9



- CodeStar
 - Wrapper around developer tools to simplify setup
 - Templates
 - Team Management
 - Central Project Dashboard
 - Free of charge
- Cloud9
 - Cloud-based full-fledged IDE



- CodeStar
 - Wrapper around developer tools to simplify setup
 - Templates
 - Team Management
 - Central Project Dashboard
 - Free of charge
- Cloud9
 - Cloud-based full-fledged IDE
 - Runs in a web browser



- CodeStar
 - Wrapper around developer tools to simplify setup
 - Templates
 - Team Management
 - Central Project Dashboard
 - Free of charge
- Cloud9
 - Cloud-based full-fledged IDE
 - Runs in a web browser
 - Collaborative editing and chat



- CodeStar
 - Wrapper around developer tools to simplify setup
 - Templates
 - Team Management
 - Central Project Dashboard
 - Free of charge
- Cloud9
 - Cloud-based full-fledged IDE
 - Runs in a web browser
 - Collaborative editing and chat
 - Greatly-integrated with AWS



■ CodeStar

- Wrapper around developer tools to simplify setup
- Templates
- Team Management
- Central Project Dashboard
- Free of charge

■ Cloud9

- Cloud-based full-fledged IDE
- Runs in a web browser
- Collaborative editing and chat
- Greatly-integrated with AWS
- Free of charge



AN AWS BESTIARIUM



MACHINE LEARNING

- Amazon SageMaker



- Amazon SageMaker
 - Preconfigured for Tensorflow, MXNet...



- Amazon SageMaker
 - Preconfigured for Tensorflow, MXNet...
 - Build, Train and Deploy phases



- Amazon SageMaker
 - Preconfigured for Tensorflow, MXNet...
 - Build, Train and Deploy phases
 - Pay based on build time, train time and hosting time



- Comprehend (for NLP) [Dashboard](#)



- Comprehend (for NLP) [Dashboard](#)
- Rekognition (Visual Analysis) [Dashboard](#)



- Comprehend (for NLP) [Dashboard](#)
- Rekognition (Visual Analysis) [Dashboard](#)
- Translate

- Comprehend (for NLP) [Dashboard](#)
- Rekognition (Visual Analysis) [Dashboard](#)
- Translate
- Polly (text-to-speech)

- Comprehend (for NLP) [Dashboard](#)
- Rekognition (Visual Analysis) [Dashboard](#)
- Translate
- Polly (text-to-speech)
- Transcribe (speech-to-text)

AN AWS BESTIARIUM



MISCELLANEA

- Cognito



- Cognito
 - Sign-up and authentication



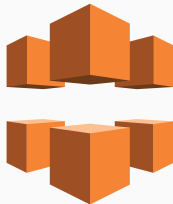
- Cognito
 - Sign-up and authentication
 - Federated identities



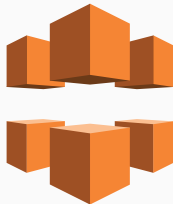
- Cognito
 - Sign-up and authentication
 - Federated identities
- CloudFront



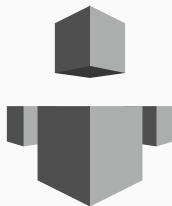
- Cognito
 - Sign-up and authentication
 - Federated identities
- CloudFront
 - Content Delivery Network



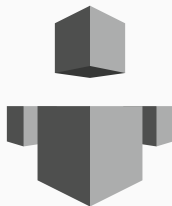
- Cognito
 - Sign-up and authentication
 - Federated identities
- CloudFront
 - Content Delivery Network
 - 116 Points of Presence in 56 cities across 24 countries



- Cognito
 - Sign-up and authentication
 - Federated identities
- CloudFront
 - Content Delivery Network
 - 116 Points of Presence in 56 cities across 24 countries
- Mechanical Turk



■ ???

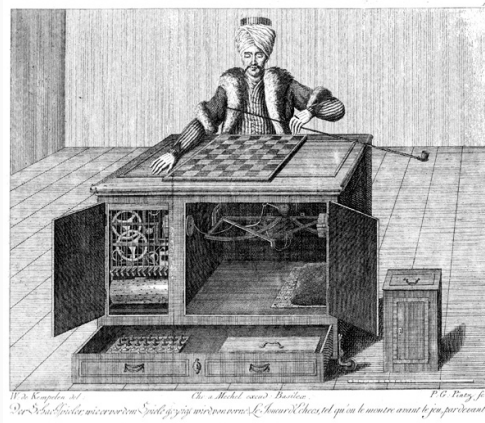


THE TURK

The Turk was a chess-playing automaton built in 1770.

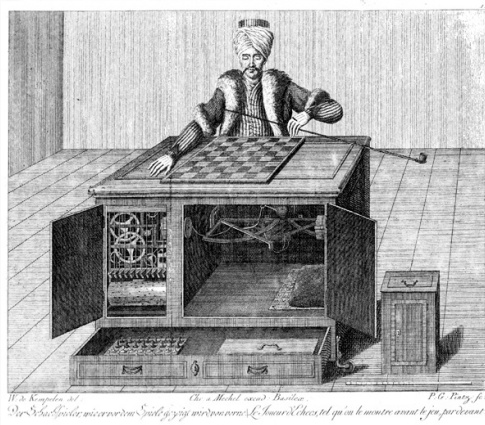
THE TURK

The Turk was a chess-playing automaton built in 1770.

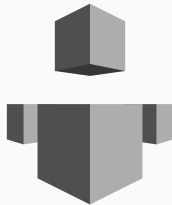


THE TURK

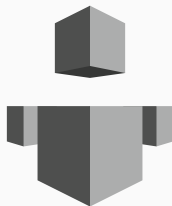
The Turk was a chess-playing automaton built in 1770. Obviously it was a fraud.



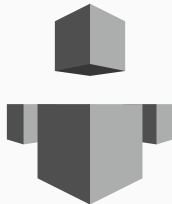
- ???



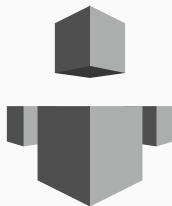
- Human Intelligence through an API



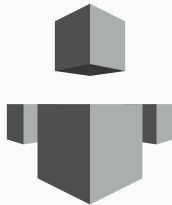
- Human Intelligence through an API
- Create HIT (Human Intelligence Task)



- Human Intelligence through an API
- Create HIT (Human Intelligence Task)
- Elastic, on-demand workforce



- Human Intelligence through an API
- Create HIT (Human Intelligence Task)
- Elastic, on-demand workforce
- Available 24/7



AN AWS BESTIARIUM



COMPUTING

- (Virtual) Servers on demand



Azure: Virtual Machines [web](#)

Google Cloud: Compute Engine [web](#)

AMAZON ELASTIC COMPUTE CLOUD (EC2)

- (Virtual) Servers on demand
- Different types of instances to suit computing needs



Azure: Virtual Machines 

Google Cloud: Compute Engine 

AMAZON ELASTIC COMPUTE CLOUD (EC2)

- (Virtual) Servers on demand
- Different types of instances to suit computing needs
- Per-second (or per-hour) billing



Azure: Virtual Machines 

Google Cloud: Compute Engine 

AMAZON ELASTIC COMPUTE CLOUD (EC2)

- (Virtual) Servers on demand
- Different types of instances to suit computing needs
- Per-second (or per-hour) billing
- Data transfer **not** included!



Azure: Virtual Machines [web](#)

Google Cloud: Compute Engine [web](#)

AMAZON ELASTIC COMPUTE CLOUD (EC2)


- (Virtual) Servers on demand
- Different types of instances to suit computing needs
- Per-second (or per-hour) billing
- Data transfer **not** included!
- Persistent storage **not** included!



Azure: Virtual Machines [web](#)

Google Cloud: Compute Engine [web](#)

AMAZON ELASTIC COMPUTE CLOUD (EC2)


- (Virtual) Servers on demand
- Different types of instances to suit computing needs
- Per-second (or per-hour) billing
- Data transfer **not** included!
- Persistent storage **not** included!
 -  EBS/EFS



Azure: Virtual Machines 

Google Cloud: Compute Engine 

AMAZON ELASTIC COMPUTE CLOUD (EC2)

- (Virtual) Servers on demand
- Different types of instances to suit computing needs
- Per-second (or per-hour) billing
- Data transfer **not** included!
- Persistent storage **not** included!
 -  EBS/EFS
- Scaling **not** included!



Azure: Virtual Machines 

Google Cloud: Compute Engine 

- *Scaling is the ability to increase or decrease the compute capacity of your application*



Azure: Virtual Machine Scale Sets 

Google Cloud: Load Balancing 

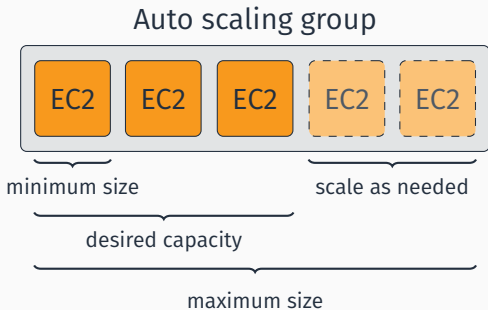
- *Scaling is the ability to increase or decrease the compute capacity of your application*
- Scale your application manually, on a scheduled basis or on demand



Azure: Virtual Machine Scale Sets 

Google Cloud: Load Balancing 

AMAZON EC2 AUTO SCALING: DETAILS



- Distributes incoming traffic across multiple EC2 instances



AMAZON ELASTIC LOAD BALANCING (ELB)

- Distributes incoming traffic across multiple EC2 instances
- Pay-per-use billing



AMAZON ELASTIC LOAD BALANCING (ELB)

- Distributes incoming traffic across multiple EC2 instances
- Pay-per-use billing
 - Execution time

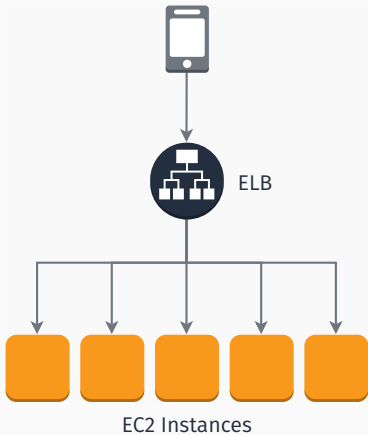


AMAZON ELASTIC LOAD BALANCING (ELB)

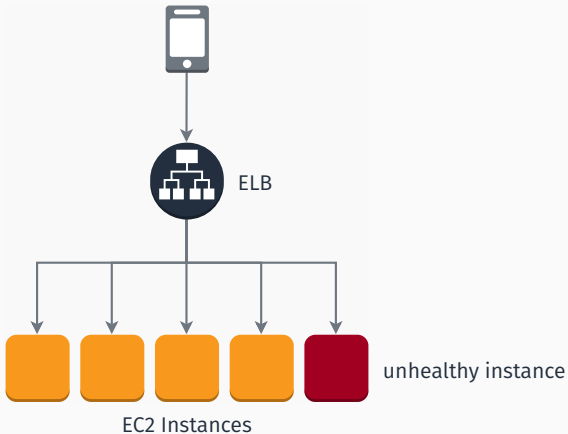
- Distributes incoming traffic across multiple EC2 instances
- Pay-per-use billing
 - Execution time
 - Number of requests / traffic



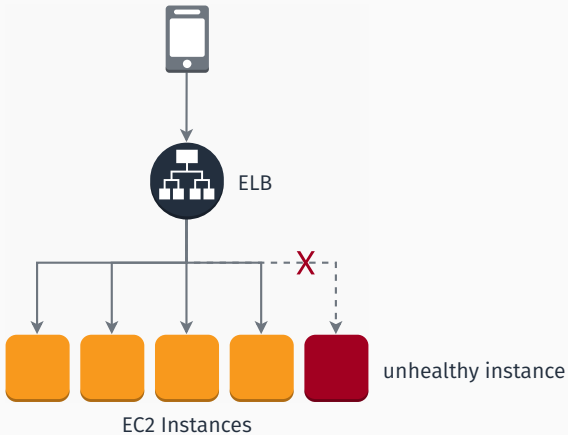
AMAZON ELASTIC LOAD BALANCING (ELB)



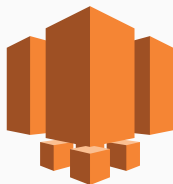
AMAZON ELASTIC LOAD BALANCING (ELB)



AMAZON ELASTIC LOAD BALANCING (ELB)



- A lightweight, simplified offer



Websites:

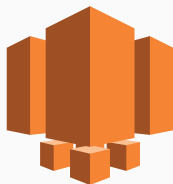


EC2



Lightsail

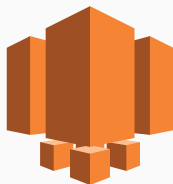
- A lightweight, simplified offer
- Bundles computing, storage, and networking capacity



Websites:



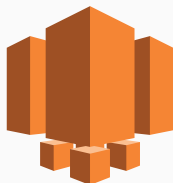
- A lightweight, simplified offer
- Bundles computing, storage, and networking capacity
- Preconfigured instances for



Websites:



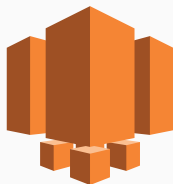
- A lightweight, simplified offer
- Bundles computing, storage, and networking capacity
- Preconfigured instances for
 - Debian, Windows Server, ...



Websites:



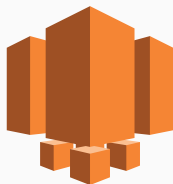
- A lightweight, simplified offer
- Bundles computing, storage, and networking capacity
- Preconfigured instances for
 - Debian, Windows Server, ...
 - Wordpress, Magento, Redmine, ...



Websites:



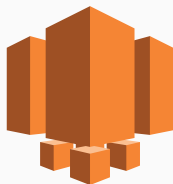
- A lightweight, simplified offer
- Bundles computing, storage, and networking capacity
- Preconfigured instances for
 - Debian, Windows Server, ...
 - Wordpress, Magento, Redmine, ...
 - LAMP stack, Nginx, ...



Websites:



- A lightweight, simplified offer
- Bundles computing, storage, and networking capacity
- Preconfigured instances for
 - Debian, Windows Server, ...
 - Wordpress, Magento, Redmine, ...
 - LAMP stack, Nginx, ...
- Low and **predictable** monthly costs



Websites:



Lightsail

- *“Easy to begin, impossible to outgrow”*



- *“Easy to begin, impossible to outgrow”*
- Easy-to-use service to deploy web apps



- *“Easy to begin, impossible to outgrow”*
- Easy-to-use service to deploy web apps
- Supports Apache, Nginx, IIS and more



- *“Easy to begin, impossible to outgrow”*
- Easy-to-use service to deploy web apps
- Supports Apache, Nginx, IIS and more
- Supports Java, .NET, PHP, Node.js, Python, Ruby, Go, and Docker



- *“Easy to begin, impossible to outgrow”*
- Easy-to-use service to deploy web apps
- Supports Apache, Nginx, IIS and more
- Supports Java, .NET, PHP, Node.js, Python, Ruby, Go, and Docker
- Manages auto-scaling, load balancing, health monitoring



- *“Easy to begin, impossible to outgrow”*
- Easy-to-use service to deploy web apps
- Supports Apache, Nginx, IIS and more
- Supports Java, .NET, PHP, Node.js, Python, Ruby, Go, and Docker
- Manages auto-scaling, load balancing, health monitoring
- Customizable



- *“Easy to begin, impossible to outgrow”*
- Easy-to-use service to deploy web apps
- Supports Apache, Nginx, IIS and more
- Supports Java, .NET, PHP, Node.js, Python, Ruby, Go, and Docker
- Manages auto-scaling, load balancing, health monitoring
- Customizable
- Free of charge. Pay only for the AWS resources you use.



A LITTLE RECAP

So far we've seen:

So far we've seen:

- Elastic Compute Cloud (EC2)

So far we've seen:

- Elastic Compute Cloud (EC2)
 - Auto-scaling, Elastic Load Balancing

So far we've seen:

- Elastic Compute Cloud (EC2)
 - Auto-scaling, Elastic Load Balancing
- Lightsail

So far we've seen:

- Elastic Compute Cloud (EC2)
 - Auto-scaling, Elastic Load Balancing
- Lightsail
- Elastic Beanstalk

So far we've seen:

- Elastic Compute Cloud (EC2)
 - Auto-scaling, Elastic Load Balancing
- Lightsail
- Elastic Beanstalk

We have to (somewhat) care about the infrastructure!

It's demo time!

- Checkout a very simple web application written in PHP

WHAT WE'RE GOING TO DO IN THIS DEMO

- Checkout a very simple web application written in PHP
- Run it locally (optional)

WHAT WE'RE GOING TO DO IN THIS DEMO

- Checkout a very simple web application written in PHP
- Run it locally (optional)
- Deploy it to the cloud using Amazon Elastic Beanstalk

WHAT WE'RE GOING TO DO IN THIS DEMO

- Checkout a very simple web application written in PHP
- Run it locally (optional)
- Deploy it to the cloud using Amazon Elastic Beanstalk
- Doable in 30 minutes at home.

We'll deploy a very simple website for this very talk. The web app has two pages:

We'll deploy a very simple website for this very talk. The web app has two pages:

- a substantially static **homepage**

We'll deploy a very simple website for this very talk. The web app has two pages:

- a substantially static **homepage**
- a **comment** page allowing users to leave feedbacks.

We'll deploy a very simple website for this very talk. The web app has two pages:

- a substantially static **homepage**
- a **comment** page allowing users to leave feedbacks.

We'll deploy a very simple website for this very talk. The web app has two pages:

- a substantially static **homepage**
- a **comment** page allowing users to leave feedbacks.

Technologies involved:

- Symfony framework

We'll deploy a very simple website for this very talk. The web app has two pages:

- a substantially static **homepage**
- a **comment** page allowing users to leave feedbacks.

Technologies involved:

- Symfony framework
- Doctrine ORM

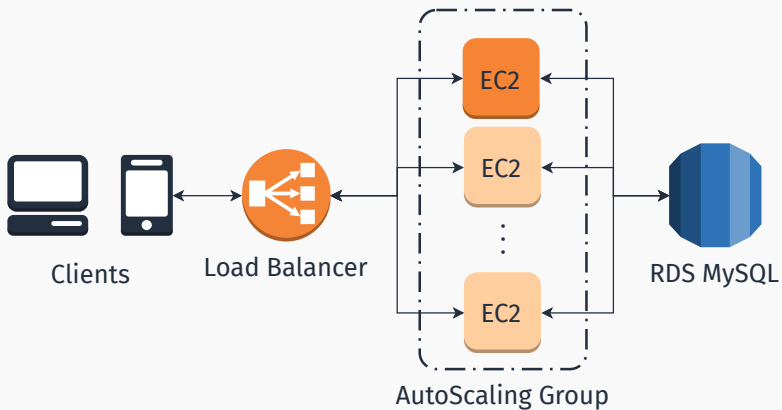
We'll deploy a very simple website for this very talk. The web app has two pages:

- a substantially static **homepage**
- a **comment** page allowing users to leave feedbacks.

Technologies involved:

- Symfony framework
- Doctrine ORM
- Webpack, Sass

ARCHITECTURE



▶▶ Skip tutorial

WHAT YOU'LL NEED

- An AWS account (a free one will suffice)

WHAT YOU'LL NEED

- An AWS account (a free one will suffice)
- `git` version control (recommended)

WHAT YOU'LL NEED

- An AWS account (a free one will suffice)
- `git` version control (recommended)
- If you want to build and run the app locally:

- An AWS account (a free one will suffice)
- `git` version control (recommended)
- If you want to build and run the app locally:
 - An AMP (Apache, PHP \geq 7.1.3, MySQL \geq 5.7) stack

- An AWS account (a free one will suffice)
- git version control (recommended)
- If you want to build and run the app locally:
 - An AMP (Apache, PHP \geq 7.1.3, MySQL \geq 5.7) stack
 - Composer package manager

- An AWS account (a free one will suffice)
- git version control (recommended)
- If you want to build and run the app locally:
 - An AMP (Apache, PHP \geq 7.1.3, MySQL \geq 5.7) stack
 - Composer package manager
 - Node.js

STEP 1: GET THE APP

Clone the git repository 

```
D:\Desktop> git clone https://github.com/luistar/  
serverful-webapp.git serverful-webapp
```

STEP 2: INSTALL DEPENDENCIES

Install dependencies with composer

```
D:\Desktop> cd serverful-webapp
```

```
D:\Desktop\serverful-webapp> composer install
```

Then install Node.js dependencies

```
D:\Desktop\serverful-webapp> npm install
```


STEP 3: CONFIGURATION PARAMETERS

Start your database instance and create an user for the webapp. Once you are done, update the configuration file `config/packages/database-config.php` accordingly.

```
2 //get parameter from environment or fallback to defaults
3 $db_host = (
4     (isset($_SERVER['RDS_HOST'])) ?
5     ($_SERVER['RDS_HOST']) : ('localhost')
6 );
7 /* And following lines */
```

In `config/services.yaml` replace the dummy text with your Google Maps API Key.

```
1 parameters:
2     locale: 'en'
3     app.gmaps_api_key: '<YOUR GMAPS API KEY HERE>'
```

STEP 4: BUILD ASSETS AND CREATE DATABASE SCHEMA

Build assets with

```
D:\Desktop\serverful-webapp> npm run webpack-dev
```

Then create the database and the data schema by running

```
D:\Desktop\serverful-webapp> npm run drop-database  
D:\Desktop\serverful-webapp> npm run create-database  
D:\Desktop\serverful-webapp> npm run create-schema
```

STEP 5: RUN THE APP

Now you can start the dev server and check out the app.

```
D:\Desktop\serverful-webapp> npm run serve
```

Once the server started, visit the webapp at localhost:8000

STEP 6: CREATE A SOURCE BUNDLE

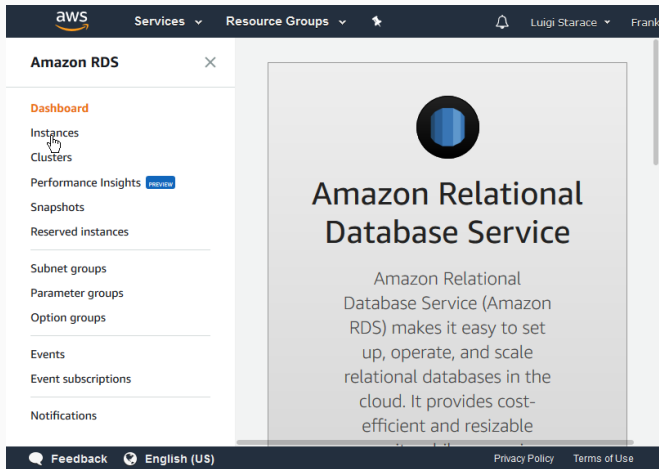
Elastic Beanstalk requires a single WAR or ZIP archive containing your app. To create a source bundle for our app, run

```
D:\Desktop\serverful-webapp> npm run create-source-  
bundle
```

A `serverful-app.zip` (our source bundle) archive will be created in the app root.

STEP 7: CREATE A DATABASE INSTANCE

Go to the RDS Console and select “instances” .



The screenshot displays the AWS Management Console interface for Amazon RDS. At the top, the AWS logo is on the left, and navigation links for 'Services', 'Resource Groups', and user information 'Luigi Starace' are on the right. The main content area is titled 'Amazon RDS' and features a sidebar on the left with a list of navigation options: 'Dashboard', 'Instances' (highlighted with a mouse cursor), 'Clusters', 'Performance Insights' (with a 'preview' button), 'Snapshots', 'Reserved instances', 'Subnet groups', 'Parameter groups', 'Option groups', 'Events', 'Event subscriptions', and 'Notifications'. The main panel shows the 'Amazon Relational Database Service' header with a blue circular icon and a descriptive paragraph: 'Amazon Relational Database Service (Amazon RDS) makes it easy to set up, operate, and scale relational databases in the cloud. It provides cost-efficient and resizable'. The footer contains 'Feedback', 'English (US)', 'Privacy Policy', and 'Terms of Use'.

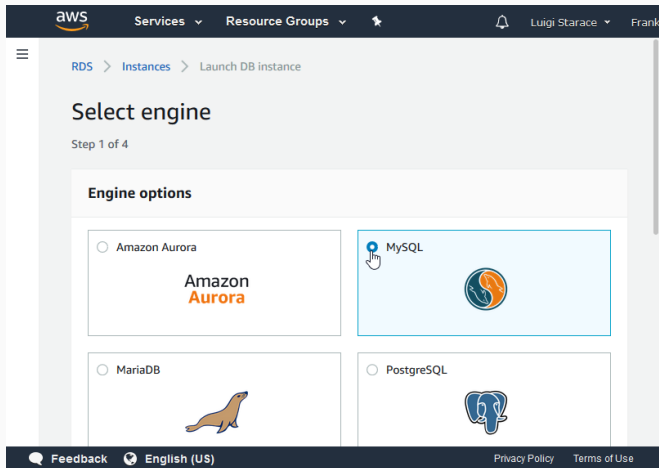
STEP 7: CREATE A DATABASE INSTANCE

Select “Launch DB instance” .

The screenshot displays the AWS Management Console interface for Amazon RDS. The top navigation bar includes the AWS logo, 'Services', 'Resource Groups', a search icon, a notification bell, and the user's name 'Luigi Starace' with a dropdown arrow. The main content area is titled 'Amazon RDS' and 'RDS > Instances'. On the left, a sidebar menu lists various RDS features: Dashboard, Instances (highlighted in orange), Clusters, Performance Insights (with a 'PREVIEW' badge), Snapshots, Reserved instances, Subnet groups, Parameter groups, Option groups, Events, Event subscriptions, and Notifications. The main panel shows 'Instances (0)' with a refresh icon, an 'Instance actions' dropdown, and a 'Restore from S3' button. A prominent orange button labeled 'Launch DB instance' is the focus, with a mouse cursor hovering over it. Below this is a search bar labeled 'Filter instances' and pagination controls showing '1' of 1 items. A table header for 'DB instance' is visible at the bottom of the main panel. The footer contains 'Feedback', 'English (US)', 'Privacy Policy', and 'Terms of Use'.

STEP 7: CREATE A DATABASE INSTANCE

Select MySQL DBMS.



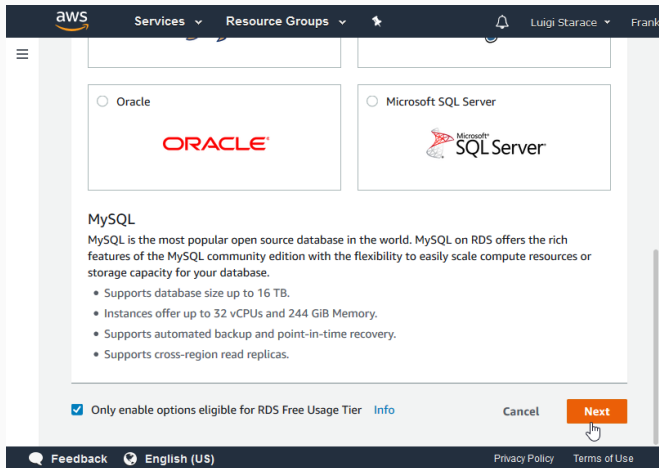
The screenshot shows the AWS Management Console interface for launching a database instance. The breadcrumb navigation indicates the path: RDS > Instances > Launch DB instance. The main heading is "Select engine", labeled as "Step 1 of 4". Under the "Engine options" section, four database engines are presented as selectable cards:

- Amazon Aurora: Features the Amazon Aurora logo.
- MySQL: Features the MySQL logo and is highlighted with a light blue border and a mouse cursor pointing to its selection radio button.
- MariaDB: Features the MariaDB logo (a seal).
- PostgreSQL: Features the PostgreSQL logo (a blue elephant).

The bottom of the console shows a dark navigation bar with "Feedback", "English (US)", "Privacy Policy", and "Terms of Use" links.

STEP 7: CREATE A DATABASE INSTANCE

Enable only free-tier options and continue.



The screenshot shows the AWS console interface for creating a database instance. At the top, the AWS logo is on the left, and navigation links for 'Services', 'Resource Groups', and user information 'Luigi Starace' and 'Frank' are on the right. Below the navigation bar, there are two radio button options: 'Oracle' and 'Microsoft SQL Server'. The 'Oracle' option is selected, and its logo is displayed in a large box. Below these options, the 'MySQL' section is visible, featuring a description of MySQL on RDS and a list of features. At the bottom of the console, there is a checkbox labeled 'Only enable options eligible for RDS Free Usage Tier' which is checked, and an 'Info' link. To the right of this are 'Cancel' and 'Next' buttons. A mouse cursor is pointing at the 'Next' button. The footer of the console includes 'Feedback', 'English (US)', 'Privacy Policy', and 'Terms of Use'.

aws Services Resource Groups Luigi Starace Frank

Oracle

ORACLE

Microsoft SQL Server

Microsoft SQL Server

MySQL

MySQL is the most popular open source database in the world. MySQL on RDS offers the rich features of the MySQL community edition with the flexibility to easily scale compute resources or storage capacity for your database.

- Supports database size up to 16 TB.
- Instances offer up to 32 vCPUs and 244 GiB Memory.
- Supports automated backup and point-in-time recovery.
- Supports cross-region read replicas.

Only enable options eligible for RDS Free Usage Tier [Info](#) Cancel Next

Feedback English (US) Privacy Policy Terms of Use

STEP 7: CREATE A DATABASE INSTANCE

Select MySQL version 5.7.21

The screenshot shows the AWS Management Console interface for launching a Database Instance. The breadcrumb navigation is RDS > Instances > Launch DB instance. The page title is 'Specify DB details', which is Step 2 of 3. Under the 'Instance specifications' section, the DB engine is set to 'MySQL Community Edition'. The 'License model' dropdown is set to 'general-public-license'. The 'DB engine version' dropdown is set to 'mysql 5.7.21'. At the bottom of the console, there are links for 'Feedback', 'English (US)', 'Privacy Policy', and 'Terms of Use'.

aws Services Resource Groups Luigi Starace Frank

RDS > Instances > Launch DB instance

Specify DB details

Step 2 of 3

Instance specifications

Estimate your monthly costs for the DB Instance using the [AWS Simple Monthly Calculator](#).

DB engine
MySQL Community Edition

License model [Info](#)
general-public-license

DB engine version [Info](#)
mysql 5.7.21

[Feedback](#) [English \(US\)](#) [Privacy Policy](#) [Terms of Use](#)

STEP 7: CREATE A DATABASE INSTANCE

Select `db.t2.micro` instance.

The screenshot shows the AWS Management Console interface for creating a new Amazon RDS database instance. At the top, the AWS logo is on the left, and navigation links for 'Services', 'Resource Groups', and user information 'Luigi Starace' and 'Frank' are on the right. A blue information box titled 'Free tier' is prominently displayed, stating: 'The Amazon RDS Free Tier provides a single db.t2.micro instance as well as up to 20 GB of storage, allowing new AWS customers to gain hands-on experience with Amazon RDS. Learn more about the RDS Free Tier and the instance restrictions [here](#). Only enable options eligible for RDS Free Usage Tier [Info](#)'. Below this, the configuration options are shown: 'DB instance class' is set to 'db.t2.micro — 1 vCPU, 1 GiB RAM'; 'Multi-AZ deployment' is set to 'No' (with 'Create replica in different zone' as an unselected option); 'Storage type' is set to 'General Purpose (SSD)'; and 'Allocated storage' is set to '20 GB'. At the bottom of the console, there are links for 'Feedback', 'English (US)', 'Privacy Policy', and 'Terms of Use'.

STEP 7: CREATE A DATABASE INSTANCE

Enter your desired settings (remember the password! ⚠️).

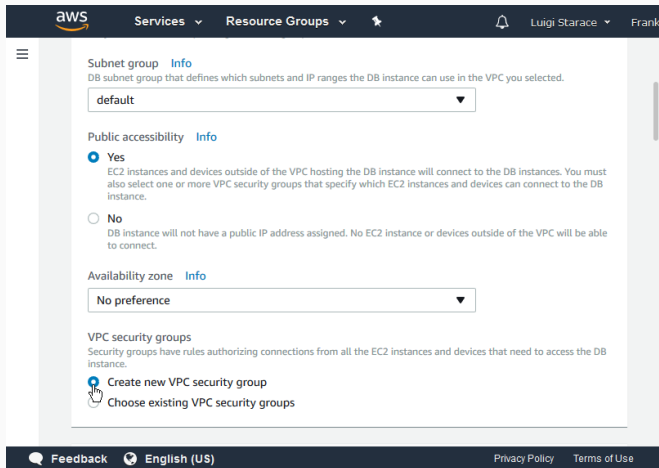
The screenshot shows the AWS console interface for configuring a database instance. The top navigation bar includes the AWS logo, 'Services', 'Resource Groups', a search icon, a notification bell, and the user's name 'Luigi Starace' with a dropdown arrow. The main content area is titled 'Settings' and contains the following fields:

- DB instance identifier** (Info): A text input field containing 'serverful-webapp-db'. Below it, a note states: 'Specify a name that is unique for all DB instances owned by your AWS account in the current region. DB Instance identifier is case insensitive, but stored as all lower-case, as in "mydbinstance". Must contain from 1 to 63 alphanumeric characters or hyphens (1 to 15 for SQL Server). First character must be a letter. Cannot end with a hyphen or contain two consecutive hyphens.'
- Master username** (Info): A text input field containing 'serverfulwebapp'. Below it, a note states: 'Specify an alphanumeric string that defines the login ID for the master user. Master Username must start with a letter. Must contain 1 to 16 alphanumeric characters.'
- Master password** (Info): A password input field with 10 dots.
- Confirm password** (Info): A password input field with 10 dots.

At the bottom of the settings area, there are three buttons: 'Cancel', 'Previous', and 'Next'. The 'Next' button is highlighted in orange and has a mouse cursor pointing to it. The footer of the console includes 'Feedback', 'English (US)', 'Privacy Policy', and 'Terms of Use'.


STEP 7: CREATE A DATABASE INSTANCE

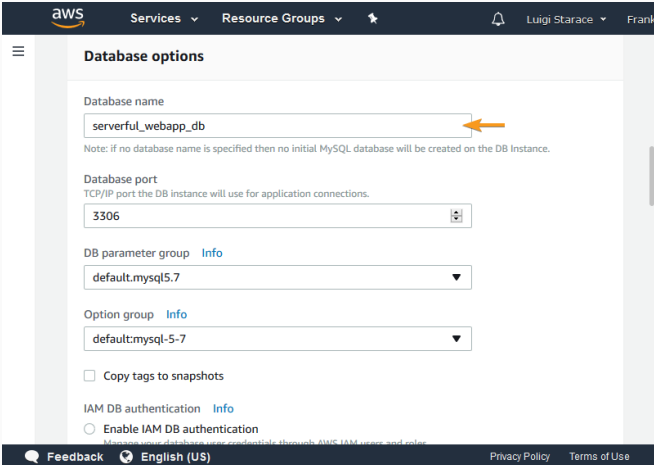
Be sure to select “create a new security group” .



The screenshot shows the AWS console configuration page for a database instance. The top navigation bar includes the AWS logo, 'Services', 'Resource Groups', a search icon, a notification bell, and the user's name 'Luigi Starace' with a dropdown arrow. A hamburger menu icon is on the left. The main content area is titled 'Subnet group Info' and includes a description: 'DB subnet group that defines which subnets and IP ranges the DB instance can use in the VPC you selected.' Below this is a dropdown menu with 'default' selected. The next section is 'Public accessibility Info', with 'Yes' selected. The 'No' option is also visible. The 'Availability zone Info' section has a dropdown menu with 'No preference' selected. The 'VPC security groups' section has a description: 'Security groups have rules authorizing connections from all the EC2 instances and devices that need to access the DB instance.' Two options are listed: 'Create new VPC security group' (which is selected with a blue circle and a mouse cursor) and 'Choose existing VPC security groups'. The bottom footer contains 'Feedback', 'English (US)', 'Privacy Policy', and 'Terms of Use'.

STEP 7: CREATE A DATABASE INSTANCE

Enter a database name for the instance (**important!** ) and leave the rest as is.



The screenshot shows the 'Database options' configuration page in the AWS Management Console. The 'Database name' field is populated with 'serverful_webapp_db' and has an orange arrow pointing to it. Below this field is a note: 'Note: if no database name is specified then no initial MySQL database will be created on the DB Instance.' Other fields include 'Database port' (3306), 'DB parameter group' (default.mysql5.7), and 'Option group' (default.mysql-5-7). There are also checkboxes for 'Copy tags to snapshots' and 'IAM DB authentication' (with 'Enable IAM DB authentication' selected).

aws Services Resource Groups Luigi Starace Frank

Database options

Database name
serverful_webapp_db

Note: if no database name is specified then no initial MySQL database will be created on the DB Instance.

Database port
TCP/IP port the DB instance will use for application connections.
3306

DB parameter group [Info](#)
default.mysql5.7

Option group [Info](#)
default.mysql-5-7

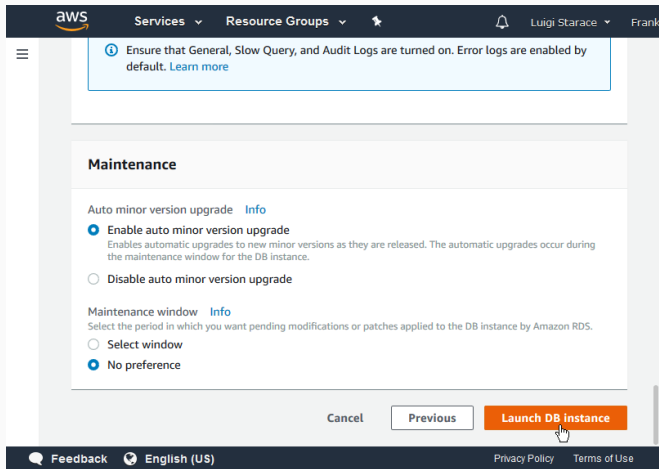
Copy tags to snapshots

IAM DB authentication [Info](#)
 Enable IAM DB authentication

Feedback English (US) Privacy Policy Terms of Use

STEP 7: CREATE A DATABASE INSTANCE

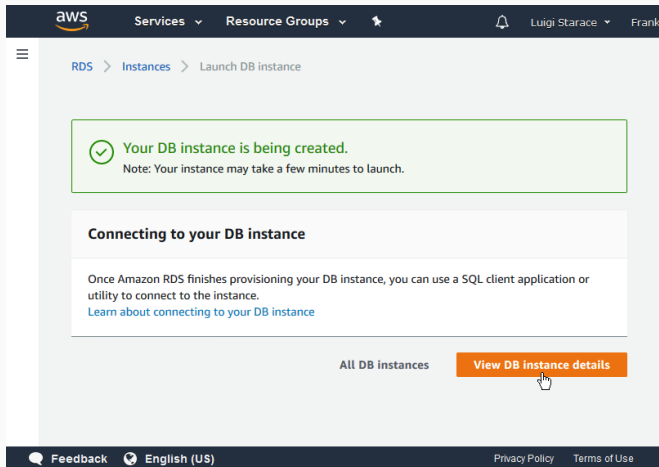
Click on “Launch DB Instance” .



The screenshot shows the AWS Management Console interface for the 'Launch DB Instance' wizard, specifically the 'Maintenance' configuration step. At the top, the AWS logo is on the left, and navigation links for 'Services', 'Resource Groups', and user information ('Luigi Starace', 'Frank') are on the right. A notification banner at the top states: 'Ensure that General, Slow Query, and Audit Logs are turned on. Error logs are enabled by default. Learn more'. The main content area is titled 'Maintenance' and contains two sections: 'Auto minor version upgrade' and 'Maintenance window'. Under 'Auto minor version upgrade', the 'Enable auto minor version upgrade' radio button is selected, with a description: 'Enables automatic upgrades to new minor versions as they are released. The automatic upgrades occur during the maintenance window for the DB instance.' The 'Disable auto minor version upgrade' option is unselected. Under 'Maintenance window', the 'No preference' radio button is selected, with a description: 'Select the period in which you want pending modifications or patches applied to the DB instance by Amazon RDS.' At the bottom of the configuration area, there are three buttons: 'Cancel', 'Previous', and 'Launch DB instance'. A mouse cursor is pointing at the 'Launch DB instance' button. The footer of the console includes 'Feedback', 'English (US)', 'Privacy Policy', and 'Terms of Use'.

STEP 7: CREATE A DATABASE INSTANCE

The creation process takes around 15 minutes. Click on “View DB Instance Details” to visit the detail page for the instance you just created.



aws Services ▾ Resource Groups ▾ Luigi Starace ▾ Frank

RDS > Instances > Launch DB instance

✓ Your DB instance is being created.
Note: Your instance may take a few minutes to launch.

Connecting to your DB instance

Once Amazon RDS finishes provisioning your DB instance, you can use a SQL client application or utility to connect to the instance.
[Learn about connecting to your DB instance](#)

All DB instances **View DB instance details**

Feedback English (US) Privacy Policy Terms of Use

STEP 7: CREATE A DATABASE INSTANCE

When done, the status in your instance detail page will change to “available” .

The screenshot shows the AWS RDS console interface for the instance 'serverful-webapp-db'. The breadcrumb navigation is 'RDS > Instances > serverful-webapp-db'. The instance name 'serverful-webapp-db' is displayed at the top right, along with an 'Instance actions' dropdown menu. Below this is a 'Summary' section with a table of instance details:

| | | | |
|------------------------|----------------------------------|---------------------------------|-----------------------------|
| Engine MySQL 5.7.21 | DB instance class db.t2.micro | DB instance status available | Pending maintenance none |
|------------------------|----------------------------------|---------------------------------|-----------------------------|

An orange arrow points to the 'available' status in the 'DB instance status' column. Below the summary is a 'CloudWatch (54)' section with a refresh button, 'Add instance to compare', 'Monitoring' dropdown, and 'Last Hour' dropdown. A legend shows 'serverful-webapp-db' with a search input field below it. At the bottom of the console, there is a pagination bar with page numbers 1 through 9 and a settings gear icon.

© 2008 - 2018, Amazon Web Services, Inc. or its affiliates. All rights reserved. [Privacy Policy](#) [Terms of Use](#)

STEP 7: CREATE A DATABASE INSTANCE

Notice a few important elements in the details section. We're going to need these later.

Details Modify

| | | | |
|--|--|---|---|
| Configurations ARN arn:aws:rds:eu-central-1:788880174327:db:serverful-webapp-db Engine MySQL 5.7.21 License Model General Public License Created Time Sun Apr 15 08:40:55 GMT+200 2018 DB Name serverful_webapp_db Username serverfulwebapp Option Group default:mysql-5-7 Parameter group default.mysql5.7 (in-sync) | Security and network Availability zone eu-central-1c VPC vpc-12e77979 Subnet group default Subnets subnet-e4184a8f subnet-97274ada subnet-f53fb788 Security groups rds-launch-wizard-1 (sg-03a2d775170d52c34) (active) Publicly accessible Yes Endpoint serverful-webapp-db.civyafoewont.eu-central-1.rds.amazonaws.com | Instance and IOPS Instance Class db.t2.micro Storage Type General Purpose (SSD) Storage 20 GB Availability and durability DB instance status available Multi AZ No Automated backups Enabled (7 Days) Latest restore time April 15, 2018 at 8:45:00 AM UTC+2 | Maintenance details Auto minor version upgrade Yes Maintenance window mon:01:28-mon:01:58 UTC (GMT) Backup window 20:55-21:25 UTC (GMT) Pending Modifications None Pending maintenance none Encryption details Encryption enabled No |
|--|--|---|---|

STEP 7: CREATE A DATABASE INSTANCE

We'll need this instance to be accessible by our web application. To do so we're going to add a new rule to allow all instances in the same security group to access the database instance.

STEP 7: CREATE A DATABASE INSTANCE

Click on the security group in the section *Security Group Rules*.

Security group rules (2)

Filter security group rules

< 1 > ⚙

| Security group | Type | Rule |
|---|--------------------|------------------|
| rds-launch-wizard-1 (sg-03a2d775170d52c34) | CIDR/IP - Inbound | 79.51.216.139/32 |
| rds-launch-wizard-1 (sg-03a2d775170d52c34) | CIDR/IP - Outbound | 0.0.0.0/0 |

STEP 7: CREATE A DATABASE INSTANCE

Select the *Inbound* tab then click on the Edit button.

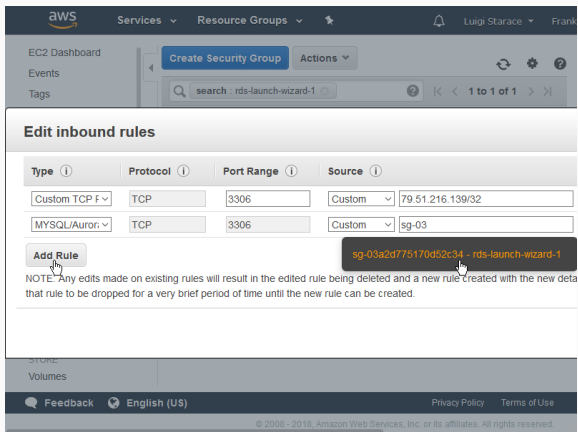
The screenshot shows the AWS Management Console interface. The top navigation bar includes the AWS logo, 'Services', 'Resource Groups', and user information 'Luigi Starace' and 'Frank'. The left sidebar contains navigation options like 'EC2 Dashboard', 'Events', 'Tags', 'Reports', 'Limits', and categories for 'INSTANCES', 'IMAGES', and 'ELASTIC BLOCK STORE'. The main content area displays the 'Create Security Group' wizard for 'rds-launch-wizard-1'. The 'Inbound' tab is selected, showing a table with one rule:

| Type | Protocol | Port Range | Source | Description |
|------------|----------|------------|----------------|-------------|
| Custom TCP | TCP | 3306 | 79.51.216.139/ | |

Below the table, there is an 'Edit' button. The 'Outbound' and 'Tags' tabs are also visible.

STEP 7: CREATE A DATABASE INSTANCE

Add a new rule as shown in the picture. Be sure to select the same security group of the database instance. Then save and return to the RDS instance detail page.



aws Services Resource Groups Luigi Starace Frank

EC2 Dashboard Create Security Group Actions

Events

Tags search: rds-launch-wizard-1

Edit inbound rules

| Type | Protocol | Port Range | Source |
|--------------|----------|------------|-------------------------|
| Custom TCP F | TCP | 3306 | Custom 79.51.216.139/32 |
| MYSQL/Auror. | TCP | 3306 | Custom sg-03 |

Add Rule sg-03a2d775170d52c34 - rds-launch-wizard-1

NOTE: Any edits made on existing rules will result in the edited rule being deleted and a new rule created with the new data that rule to be dropped for a very brief period of time until the new rule can be created.

Feedback English (US) Privacy Policy Terms of Use

© 2008 - 2018, Amazon Web Services, Inc. or its affiliates. All rights reserved.

STEP 7: CREATE A DATABASE INSTANCE

The rule you just added should be displayed among the other two.

Security group rules (3)

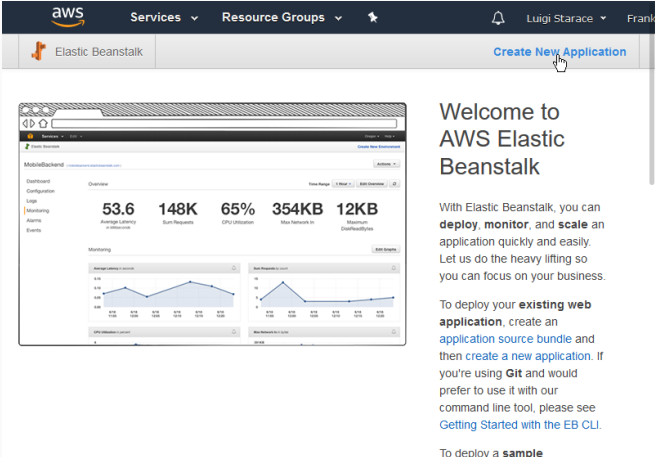
Filter security group rules

< 1 > ⚙

| Security group | Type | Rule |
|---|--------------------------|----------------------|
| rds-launch-wizard-1 (sg-03a2d775170d52c34) | CIDR/IP - Inbound | 79.51.216.139/32 |
| rds-launch-wizard-1 (sg-03a2d775170d52c34) | Security Group - Inbound | sg-03a2d775170d52c34 |
| rds-launch-wizard-1 (sg-03a2d775170d52c34) | CIDR/IP - Outbound | 0.0.0.0/0 |

STEP 7: CREATE A BEANSTALK APPLICATION

Go to the Beanstalk console and select *Create New Application*.



The screenshot shows the AWS Elastic Beanstalk console. At the top, the AWS logo is on the left, and navigation menus for 'Services', 'Resource Groups', and user information 'Luigi Starace' and 'Frank' are on the right. Below the navigation bar, the 'Elastic Beanstalk' section is active, and a blue button labeled 'Create New Application' is highlighted with a mouse cursor. Below this, a preview of the Elastic Beanstalk console interface is shown. The preview includes a 'MobileBackend' application with a 'Dashboard' and 'Monitoring' sections. The dashboard displays key metrics: Average Latency (53.6), Sum Requests (148K), CPU Utilization (65%), Max Network In (354KB), and Maximum DisposedBytes (12KB). The monitoring section contains several line graphs for Average Latency, Sum Requests, CPU Utilization, and Max Network In.

Welcome to AWS Elastic Beanstalk

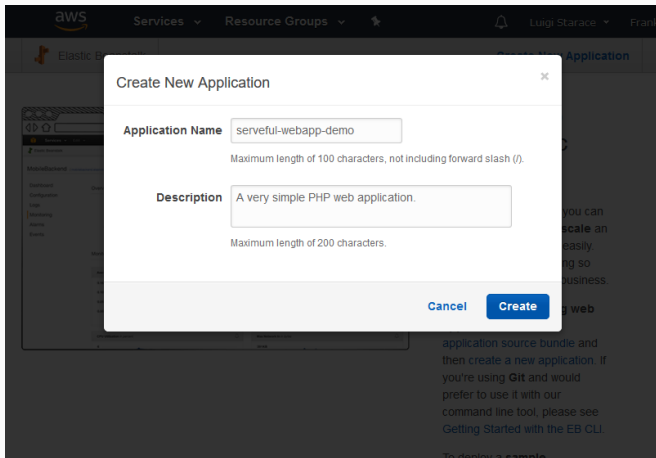
With Elastic Beanstalk, you can **deploy, monitor, and scale** an application quickly and easily. Let us do the heavy lifting so you can focus on your business.

To deploy your **existing web application**, create an [application source bundle](#) and then [create a new application](#). If you're using **Git** and would prefer to use it with our command line tool, please see [Getting Started with the EB CLI](#).

To deploy a **sample**

STEP 7: CREATE A BEANSTALK APPLICATION

Fill the form with your application information and continue.



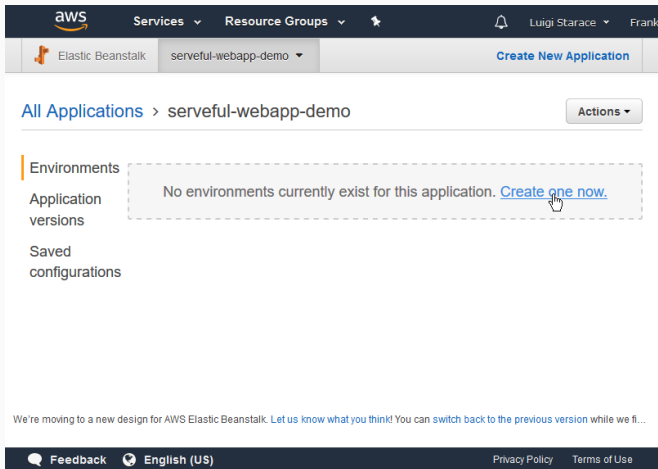
The screenshot shows the AWS Management Console interface with a modal dialog box titled "Create New Application". The dialog box contains two input fields: "Application Name" and "Description".

Application Name: The input field contains the text "serveful-webapp-demo". Below the field, a note states: "Maximum length of 100 characters, not including forward slash (/)." The background of the console shows the "Elastic Beanstalk" page with a sidebar on the left and a main content area on the right.

Description: The input field contains the text "A very simple PHP web application.". Below the field, a note states: "Maximum length of 200 characters." At the bottom right of the dialog box, there are two buttons: "Cancel" and "Create".

STEP 7: CREATE A BEANSTALK APPLICATION

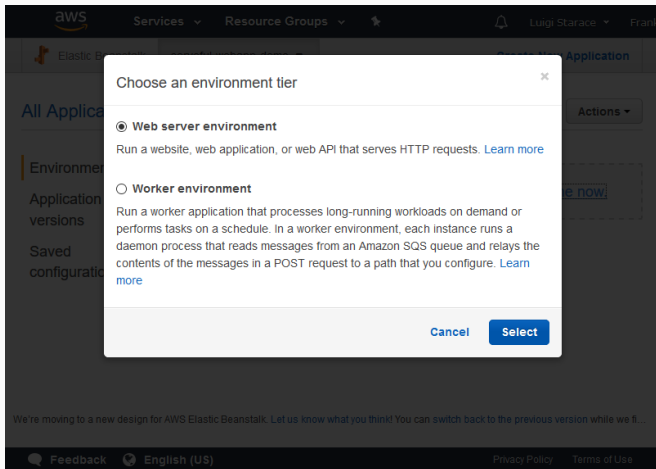
Then select *Create one now* to create a new environment for your application.



The screenshot shows the AWS Elastic Beanstalk console interface. At the top, there is a navigation bar with the AWS logo, 'Services', 'Resource Groups', a search icon, a notification bell, and user information 'Luigi Starace' and 'Frank'. Below this, a breadcrumb trail shows 'Elastic Beanstalk' and 'serveful-webapp-demo'. A 'Create New Application' button is visible in the top right. The main content area shows 'All Applications > serveful-webapp-demo' with an 'Actions' dropdown. Under the 'Environments' section, a message states 'No environments currently exist for this application. [Create one now.](#)' with a mouse cursor pointing to the link. Other sections like 'Application versions', 'Saved configurations', and a footer with 'Feedback', 'English (US)', 'Privacy Policy', and 'Terms of Use' are also visible.

STEP 7: CREATE A BEANSTALK APPLICATION

Select *Web Server Environment*, as we are going to deploy a web application.



STEP 7: CREATE A BEANSTALK APPLICATION

Fill the form with information about your environment.



Create a new environment

Launch an environment with a sample application or your own code. By creating an environment, you allow AWS Elastic Beanstalk to manage AWS resources and permissions on your behalf. [Learn more](#)

Environment information

Choose the name, subdomain, and description for your environment. These cannot be changed later.

Application name serveful-webapp-demo

Environment name

Domain

serveful-webapp.eu-central-1.elasticbeanstalk.com **is available.**

Description

STEP 7: CREATE A BEANSTALK APPLICATION

Select PHP as preconfigured platform and upload the source bundle you previously prepared.

Tier Web Server ([Choose tier](#))

Platform Preconfigured platform
Platforms published and maintained by AWS Elastic Beanstalk.

→

Custom platform ^{NEW}
Platforms created and owned by you. [Learn more](#)

Application code Sample application
Get started right away with sample code.

Existing version
Application versions that you have uploaded for `serveful-webapp-demo`.

Upload your code
→ Upload a source bundle from your computer or copy one from Amazon S3.

`serveful-webapp-demo-source`

STEP 7: CREATE A BEANSTALK APPLICATION

Select *Configure More Options* and continue.

The screenshot shows the 'Configure More Options' step in the AWS Elastic Beanstalk console. At the top, there is a dropdown menu with the text '-- Choose a custom platform --'. Below this, the 'Application code' section is active, with three radio button options: 'Sample application', 'Existing version', and 'Upload your code'. The 'Upload your code' option is selected. Under 'Existing version', there is another dropdown menu with the text '-- Choose a version --'. Under 'Upload your code', there is an 'Upload' button with a download icon and the text 'serveful-webapp-demo-source' followed by a pencil icon. At the bottom of the configuration area, there are three buttons: 'Cancel', 'Configure more options' (which has a mouse cursor over it), and 'Create environment'. Below the buttons, there is a message: 'We're moving to a new design for AWS Elastic Beanstalk. [Let us know what you think!](#) You can [switch back to the previous version](#) while we fi...'. At the very bottom, there is a dark blue footer bar with 'Feedback', 'English (US)', 'Privacy Policy', and 'Terms of Use'.

-- Choose a custom platform --

Application code

Sample application
Get started right away with sample code.

Existing version
Application versions that you have uploaded for `serveful-webapp-demo`.

-- Choose a version --

Upload your code
Upload a source bundle from your computer or copy one from Amazon S3.

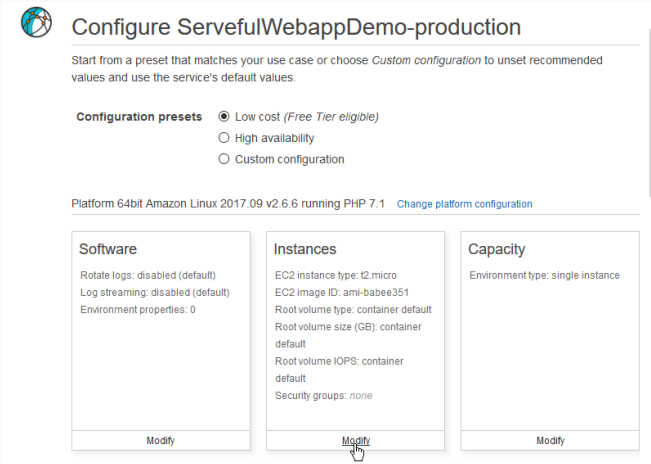
serveful-webapp-demo-source

We're moving to a new design for AWS Elastic Beanstalk. [Let us know what you think!](#) You can [switch back to the previous version](#) while we fi...


Feedback English (US) [Privacy Policy](#) [Terms of Use](#)

STEP 7: CREATE A BEANSTALK APPLICATION

In the configuration page, select *modify* with the *Instances* card.



The screenshot shows the AWS Beanstalk configuration interface. At the top, there is a globe icon and the title "Configure ServefulWebappDemo-production". Below the title, a paragraph states: "Start from a preset that matches your use case or choose *Custom configuration* to unset recommended values and use the service's default values." Underneath, the "Configuration presets" section has three radio buttons: "Low cost (Free Tier eligible)" (which is selected), "High availability", and "Custom configuration". Below this, the platform is identified as "Platform 64bit Amazon Linux 2017.09 v2.6.6 running PHP 7.1" with a link to "Change platform configuration". The main content area is divided into three cards: "Software" (with options for log rotation and streaming), "Instances" (with details on EC2 instance type, image ID, and volume settings), and "Capacity" (with environment type). Each card has a "Modify" button at the bottom. A mouse cursor is pointing at the "Modify" button of the "Instances" card.

 **Configure ServefulWebappDemo-production**

Start from a preset that matches your use case or choose *Custom configuration* to unset recommended values and use the service's default values.

Configuration presets

- Low cost (*Free Tier eligible*)
- High availability
- Custom configuration

Platform 64bit Amazon Linux 2017.09 v2.6.6 running PHP 7.1 [Change platform configuration](#)

Software

Rotate logs: disabled (default)
Log streaming: disabled (default)
Environment properties: 0

Modify

Instances

EC2 instance type: t2.micro
EC2 image ID: ami-babe0351
Root volume type: container default
Root volume size (GB): container default
Root volume IOPS: container default
Security groups: none

Modify

Capacity

Environment type: single instance

Modify

STEP 7: CREATE A BEANSTALK APPLICATION

In the instances configuration page, add the t2 instance to the same security group as the DB instance. Then save and continue.

Input/output operations per second for a provisioned IOPS (SSD) volume.

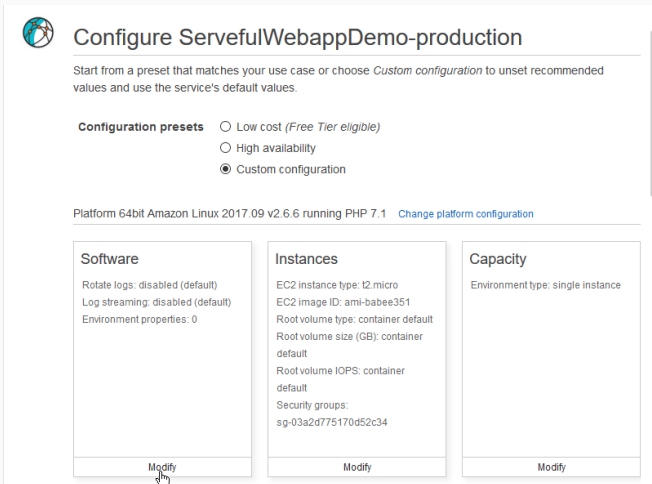
EC2 security groups

| | Group name | Group ID | Name |
|-------------------------------------|---------------------|----------------------|------|
| <input type="checkbox"/> | default | sg-c6660dab | |
| <input type="checkbox"/> | rds-launch-wizard | sg-09c29f2dc61f55e1a | |
| <input checked="" type="checkbox"/> | rds-launch-wizard-1 | sg-03a2d775170d52c34 | |


Cancel Save

STEP 7: CREATE A BEANSTALK APPLICATION

In the configuration page, select *modify* with the *Software* card.



The screenshot shows the AWS Beanstalk configuration page for an application named "ServefulWebappDemo-production". At the top left is a globe icon. The title "Configure ServefulWebappDemo-production" is followed by a horizontal line. Below the line is a paragraph: "Start from a preset that matches your use case or choose *Custom configuration* to unset recommended values and use the service's default values." Underneath is the "Configuration presets" section with three radio buttons: "Low cost (Free Tier eligible)", "High availability", and "Custom configuration" (which is selected). Below this is the platform information: "Platform 64bit Amazon Linux 2017.09 v2.6.6 running PHP 7.1" with a link "Change platform configuration". The main content area is divided into three cards: "Software", "Instances", and "Capacity". The "Software" card lists "Rotate logs: disabled (default)", "Log streaming: disabled (default)", and "Environment properties: 0". The "Instances" card lists "EC2 instance type: t2.micro", "EC2 image ID: ami-babee351", "Root volume type: container default", "Root volume size (GB): container default", "Root volume IOPS: container default", and "Security groups: sg-03a2d775170d52c34". The "Capacity" card lists "Environment type: single instance". Each card has a "Modify" button at the bottom. A mouse cursor is pointing at the "Modify" button of the "Software" card.

 **Configure ServefulWebappDemo-production**

Start from a preset that matches your use case or choose *Custom configuration* to unset recommended values and use the service's default values.

Configuration presets

- Low cost (*Free Tier eligible*)
- High availability
- Custom configuration

Platform 64bit Amazon Linux 2017.09 v2.6.6 running PHP 7.1 [Change platform configuration](#)

| Software | Instances | Capacity |
|---|---|-----------------------------------|
| Rotate logs: disabled (default) Log streaming: disabled (default) Environment properties: 0 | EC2 instance type: t2.micro EC2 image ID: ami-babee351 Root volume type: container default Root volume size (GB): container default Root volume IOPS: container default Security groups: sg-03a2d775170d52c34 | Environment type: single instance |
| Modify | Modify | Modify |

STEP 7: CREATE A BEANSTALK APPLICATION

Enter “/public” as the document root and scroll down.



Modify software

Container Options

The following settings control container behavior and let you pass key-value pairs in as OS environment variables. [Learn more](#)

Document root



The child directory of your project that acts as the public facing web root. If your root document is stored in your project directory, leave this set to /. If your root document is in a child directory (e.g., /public), set this value to match the child directory. Values should begin with a / character, and may NOT begin with a . (period).


Memory limit

The amount of memory allocated to the PHP environment. This value is written to the php.ini file.

**Zlib output
compression**

Whether PHP should use compression for output. This value is written to the php.ini file.

STEP 7: CREATE A BEANSTALK APPLICATION

Enter the required environment parameters as show in the picture. Be careful, deployment might fail if you mess up! 

Environment properties

The following properties are passed in the application as environment properties. [Learn more](#)

| Name | Value |
|---|---|
| <input type="text" value="APP_ENV"/> | <input type="text" value="prod"/> ✕ |
| <input type="text" value="RDS_HOST"/> | <input type="text" value="-central-1.rds.amazonaws.com"/> ✕ |
| <input type="text" value="RDS_NAME"/> | <input type="text" value="serverful_webapp_db"/> ✕ |
| <input type="text" value="RDS_USER"/> | <input type="text" value="serverfulwebapp"/> ✕ |
| <input type="text" value="RDS_PASSWORD"/> | <input type="text" value="██████████"/> ✕ |
| <input type="text" value="RDS_PORT"/> | <input type="text" value="3306"/> ✕ |
| <input type="text"/> | <input type="text"/> |

[Cancel](#) [Save](#)

STEP 7: CREATE A BEANSTALK APPLICATION

Click on *Create Environment* and continue.

Modify

Modify

Modify

Database

Engine: --
Instance class: --
Storage (GB): --
Multi-AZ: --

Modify

Tags

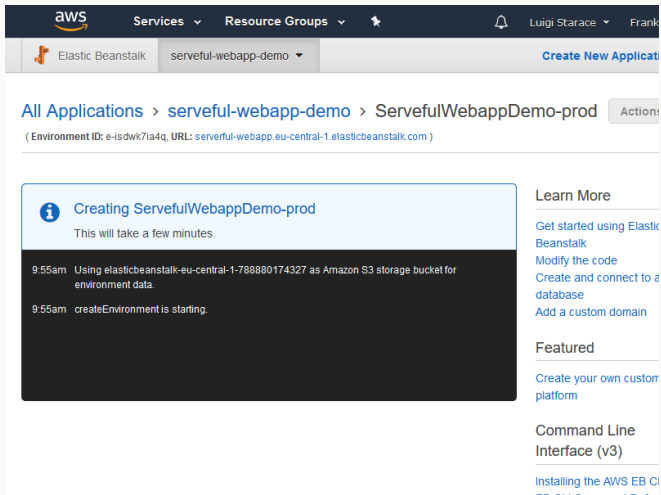
Tags: none

Modify

Cancel Previous **Create environment**

STEP 7: CREATE A BEANSTALK APPLICATION

Wait for the environment to be created. This takes about 10 minutes.



The screenshot shows the AWS Management Console interface. At the top, the navigation bar includes the AWS logo, 'Services', 'Resource Groups', a user icon, a notification bell, and the user name 'Luigi Starace'. Below the navigation bar, the breadcrumb trail reads 'Elastic Beanstalk > serveful-webapp-demo > Create New Application'. The main content area shows the path 'All Applications > serveful-webapp-demo > ServefulWebappDemo-prod' with an 'Actions' button. Below this, a message box titled 'Creating ServefulWebappDemo-prod' states 'This will take a few minutes.' and displays a log of events: '9:55am Using elasticbeanstalk-eu-central-1-788880174327 as Amazon S3 storage bucket for environment data.' and '9:55am createEnvironment is starting.' To the right of the message box, there are sections for 'Learn More' (with links for 'Get started using Elastic Beanstalk', 'Modify the code', 'Create and connect to a database', and 'Add a custom domain'), 'Featured' (with a link for 'Create your own custom platform'), and 'Command Line Interface (v3)' (with a link for 'Installing the AWS EB CLI').

STEP 8: LOAD BALANCING

Right now we have our application running on a single (virtual) web server. That's not scaling at all. Let's take advantage of the cloud and make the web application load balanced.

STEP 8: LOAD BALANCING

Select the environment's configuration view, then select the Capacity card.

Configuration overview

| | | |
|--|---|---|
| Software Rotate logs: disabled (default) Log streaming: disabled (default) Environment properties: 6 Modify | Instances EC2 instance type: t2.micro EC2 image ID: ami-babee351 Monitoring interval: 5 minute Root volume type: container default Root volume size (GB): container default Root volume IOPS: container default Security groups: sg-03a2d775170d52c34, sg-03c173ae822a52c04 Modify | Capacity Environment type: single instance Modify |
| Load balancer <i>This configuration does not contain a load balancer.</i> | Rolling updates and deployments Deployment policy: All at once Rolling updates: disabled Health check: enabled Modify | Security Service role: aws-elasticbeanstalk-service-role Virtual machine key pair: -- Virtual machine instance profile: aws-elasticbeanstalk-ec2-role Modify |

STEP 8: LOAD BALANCING

Select “Load balanced” as the environment type and customize the Auto Scaling Group.

Modify capacity

Auto Scaling Group

Configure the compute capacity of your environment and Auto Scaling settings to optimize the number of instances used.

Environment type Load balanced

Instances Min 1 Max 2

Availability Zones Any

Number of Availability Zones (AZs) to use.

Placement eu-central-1a
eu-central-1b
eu-central-1c

Specify Availability Zones (AZs) to use.

Scaling cooldown 360 seconds

STEP 8: LOAD BALANCING

Select some triggers (you can even setup time based ones), then save your changes.

Scaling triggers

Metric CPUUtilization

Change the metric that is monitored to determine if the environment's capacity is too low or too high.

Statistic Average

Choose how the metric is interpreted.

Unit Percent

Period 5 Min

The period between metric evaluations.

Breach duration 5 Min

The amount of time a metric can exceed a threshold before triggering a scaling operation.

Upper threshold 100 Percent

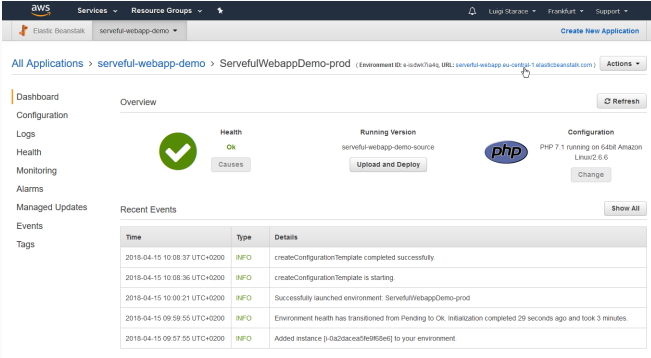
Scale up increment 1 EC2 instances

Lower threshold 90 Percent

Scale down increment -1 EC2 instances

STEP 9: ENJOY YOUR WEB APP

When it's done you should see something like this. Click on the URL to visit the load-balanced web application you just deployed on Beanstalk!



The screenshot shows the AWS Management Console interface for an Elastic Beanstalk environment named 'servful-webapp-demo'. The environment is 'ServfulWebappDemo-prod' and is in a healthy state. The console displays various metrics and configuration details.

Overview

- Health:** OK (indicated by a green checkmark icon). A 'Causes' button is available.
- Running Version:** servful-webapp-demo-source. An 'Upload and Deploy' button is present.
- Configuration:** PHP 7.1 running on 64bit Amazon Linux/2.6.6. A 'Change' button is available.

Recent Events

| Time | Type | Details |
|------------------------------|------|---|
| 2018-04-15 10:08:37 UTC+0200 | INFO | createConfigurationTemplate completed successfully. |
| 2018-04-15 10:08:36 UTC+0200 | INFO | createConfigurationTemplate is starting. |
| 2018-04-15 10:00:21 UTC+0200 | INFO | Successfully launched environment: ServfulWebappDemo-prod |
| 2018-04-15 09:59:55 UTC+0200 | INFO | Environment health has transitioned from Pending to Ok. Initialization completed 29 seconds ago and took 3 minutes. |
| 2018-04-15 09:57:55 UTC+0200 | INFO | Added instance [i-0a2d3acea0fe968e6] to your environment. |

STEP 9: ENJOY YOUR WEB APP

Sweet!



The screenshot shows a web application interface. At the top, there is a dark navigation bar with the text 'GettingToKnowAWS' in white, followed by 'Home', 'Comments', and 'Pricing' in a lighter color. Below the navigation bar is a large blue banner with white clouds. The main text on the banner reads 'Eew! What do you mean you don't know AWS?' in a large, bold, dark font. Below this, in a smaller font, it says 'Don't worry, we've got you covered!'. A dark button with the text 'Learn more' is positioned below the banner. At the bottom of the page, there is a white section with the text 'Get to know Amazon Web Services with our talk'.

GettingToKnowAWS Home Comments Pricing

Eew! What do you mean you don't know AWS?

Don't worry, we've got you covered!

[Learn more](#)

Get to know Amazon Web Services with our talk

LET'S GET BACK TO COMPUTING SERVICES

- You provide the code and say when to run it.



- You provide the code and say when to run it.
- Execution is triggered by events



- You provide the code and say when to run it.
- Execution is triggered by events
 - S3, Cognito, DynamoDB



- You provide the code and say when to run it.
- Execution is triggered by events
 - S3, Cognito, DynamoDB
 - CodeCommit, Scheduled Event



- You provide the code and say when to run it.
- Execution is triggered by events
 - S3, Cognito, DynamoDB
 - CodeCommit, Scheduled Event
- Support for Java, Node.js, C# e Python (more to come).



- You provide the code and say when to run it.
- Execution is triggered by events
 - S3, Cognito, DynamoDB
 - CodeCommit, Scheduled Event
- Support for Java, Node.js, C# e Python (more to come).
- Pay only for **actual** execution time.



- You provide the code and say when to run it.
- Execution is triggered by events
 - S3, Cognito, DynamoDB
 - CodeCommit, Scheduled Event
- Support for Java, Node.js, C# e Python (more to come).
- Pay only for **actual** execution time.
- Run your code without thinking about infrastructure



- You provide the code and say when to run it.
- Execution is triggered by events
 - S3, Cognito, DynamoDB
 - CodeCommit, Scheduled Event
- Support for Java, Node.js, C# e Python (more to come).
- Pay only for **actual** execution time.
- Run your code without thinking about infrastructure
 - No need to worry about provisioning, load balancing, scaling...



AWS Lambda imposes some limits

- Max 300 seconds execution time.



AWS Lambda imposes some limits

- Max 300 seconds execution time.
- Max 3008 MB memory allocation.



AWS Lambda imposes some limits

- Max 300 seconds execution time.
- Max 3008 MB memory allocation.
- Deployment package must be smaller than 50 MB (negotiable).



AWS Lambda imposes some limits

- Max 300 seconds execution time.
- Max 3008 MB memory allocation.
- Deployment package must be smaller than 50 MB (negotiable).
- No more than 10000 concurrent invocation of a Lambda function in a given region (negotiable).



AWS Lambda imposes some limits

- Max 300 seconds execution time.
- Max 3008 MB memory allocation.
- Deployment package must be smaller than 50 MB (negotiable).
- No more than 10000 concurrent invocation of a Lambda function in a given region (negotiable).
- For a complete list: [📄 Lambda docs](#)



FaaS (Functions as a Service)

- Functions are the unit of deployment

FaaS (Functions as a Service)

- Functions are the unit of deployment
- Executed in ephemeral, stateless containers

FaaS (Functions as a Service)

- Functions are the unit of deployment
- Executed in ephemeral, stateless containers
- Event driven

FaaS (Functions as a Service)

- Functions are the unit of deployment
- Executed in ephemeral, stateless containers
- Event driven
- No provisioning, scales automatically

FaaS (Functions as a Service)

- Functions are the unit of deployment
- Executed in ephemeral, stateless containers
- Event driven
- No provisioning, scales automatically
- Azure: Functions [▶ web](#)

FaaS (Functions as a Service)

- Functions are the unit of deployment
- Executed in ephemeral, stateless containers
- Event driven
- No provisioning, scales automatically
- Azure: Functions [▶ web](#)
- Google Cloud: Functions [▶ web](#)

FaaS (Functions as a Service)

- Functions are the unit of deployment
- Executed in ephemeral, stateless containers
- Event driven
- No provisioning, scales automatically
- Azure: Functions [▶ web](#)
- Google Cloud: Functions [▶ web](#)
- IBM: Cloud Functions [▶ web](#)

FaaS (Functions as a Service)

- Functions are the unit of deployment
- Executed in ephemeral, stateless containers
- Event driven
- No provisioning, scales automatically
- Azure: Functions [▶ web](#)
- Google Cloud: Functions [▶ web](#)
- IBM: Cloud Functions [▶ web](#)
 - Based on Apache OpenWhisk [▶ web](#)

- Orchestrating Lambda functions



- Orchestrating Lambda functions
- Define a state machine



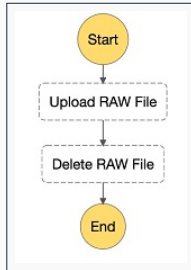


Figure 3: Sequential steps, from [AWS]

AMAZON STEP FUNCTIONS: SAMPLE II

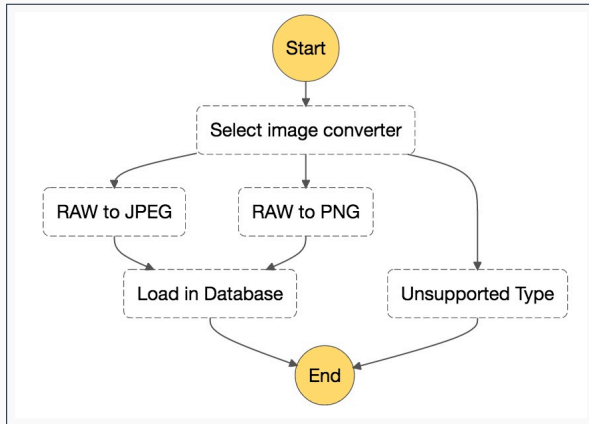


Figure 4: Branching, from [AWS]

AMAZON STEP FUNCTIONS: SAMPLE III

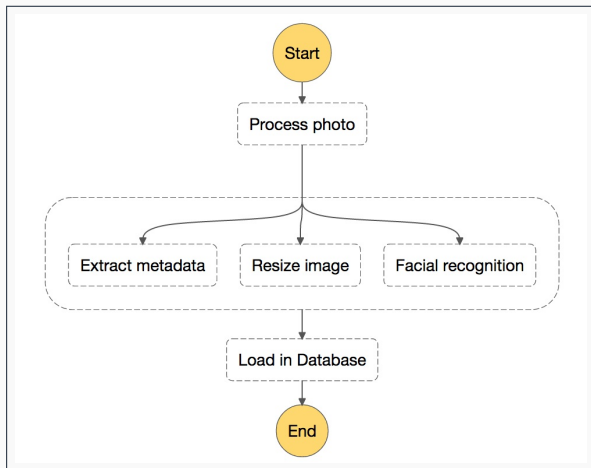


Figure 5: Parallel execution, from [AWS]

AMAZON STEP FUNCTIONS: SAMPLE IV

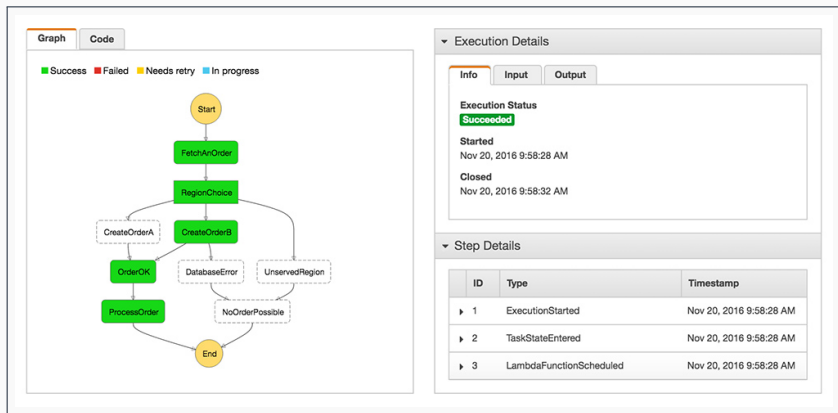


Figure 6: Monitoring executions, from [AWS]

- Create, publish, and secure APIs at any scale



- Create, publish, and secure APIs at any scale
- Authorizers (Cognito)



Serverless?

What's all the **FaaS** about?

SERVERLESS TREND

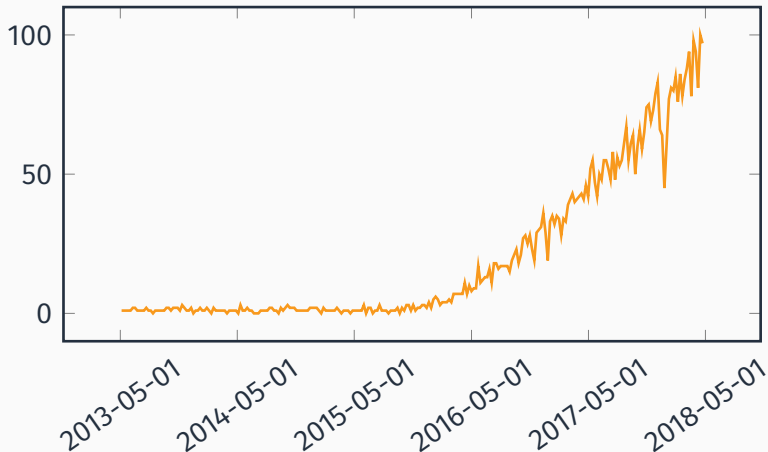


Figure 7: Last five years trend on Google for “serverless”

- No, they're not *actually* serverless...

- No, they're not *actually* serverless...
- Rely on FaaS and third-party services so that traditional always-on servers are no longer needed

- No, they're not *actually* serverless...
- Rely on FaaS and third-party services so that traditional always-on servers are no longer needed
- No worries about provisioning and scaling

- No, they're not *actually* serverless...
- Rely on FaaS and third-party services so that traditional always-on servers are no longer needed
- No worries about provisioning and scaling
- “Smarter” clients

SERVERLESS USE CASES: SPORADIC REQUESTS

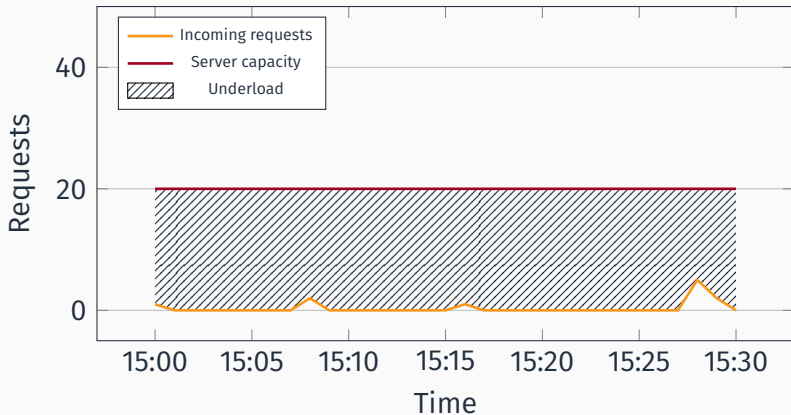


Figure 8: Sporadic requests example

SERVERLESS USE CASES: INCONSISTENT REQUESTS

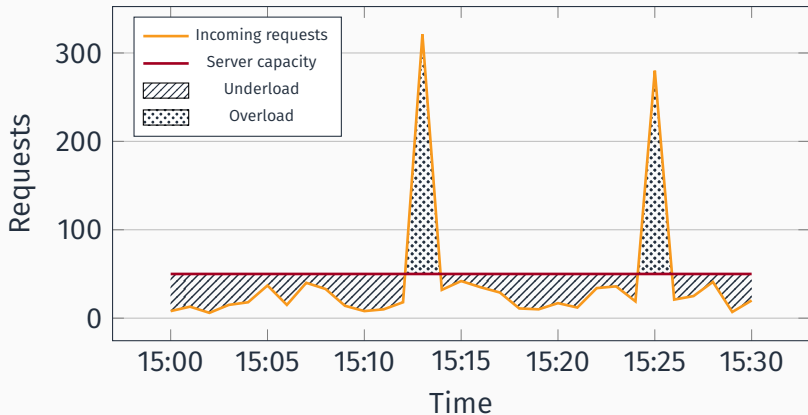


Figure 9: Inconsistent requests example


Pros

- Reduce costs 


Pros

- Reduce costs 
- No worries about provisioning, scaling

Pros


- Reduce costs 
- No worries about provisioning, scaling
- Less time to market

Pros

- Reduce costs 
- No worries about provisioning, scaling
- Less time to market

Cons


Pros

- Reduce costs 
- No worries about provisioning, scaling
- Less time to market

Cons

- Limits


Pros

- Reduce costs 
- No worries about provisioning, scaling
- Less time to market

Cons

- Limits
- Vendor lock-in

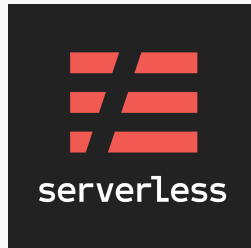
Pros


- Reduce costs 
- No worries about provisioning, scaling
- Less time to market

Cons

- Limits
- Vendor lock-in
- Testing

- Serverless Framework 





- Serverless Framework  “toolkit to deploy and operate serverless architecture” .






- Serverless Framework  [web](#)
 - “toolkit to deploy and operate serverless architecture” .
 - Works with AWS, Google, Microsoft, IBM.



- Serverless Framework 
 - “toolkit to deploy and operate serverless architecture” .
 - Works with AWS, Google, Microsoft, IBM.
- APEX 






SERVERLESS ARCHITECTURES: TOOLS

- Serverless Framework 
 - “toolkit to deploy and operate serverless architecture” .
 - Works with AWS, Google, Microsoft, IBM.
- APEX 
- AWS SAM 






SERVERLESS ARCHITECTURES: TOOLS

- Serverless Framework  web
 - “toolkit to deploy and operate serverless architecture” .
 - Works with AWS, Google, Microsoft, IBM.
- APEX  web
- AWS SAM  web
 - Serverless Application Model






SERVERLESS ARCHITECTURES: TOOLS

- Serverless Framework 
 - “toolkit to deploy and operate serverless architecture” .
 - Works with AWS, Google, Microsoft, IBM.
- APEX 
- AWS SAM 
 - Serverless Application Model
 - “Define serverless applications with a simple and clean syntax”



SERVERLESS ARCHITECTURES: TOOLS

- Serverless Framework 
 - “toolkit to deploy and operate serverless architecture” .
 - Works with AWS, Google, Microsoft, IBM.
- APEX 
- AWS SAM 
 - Serverless Application Model
 - “Define serverless applications with a simple and clean syntax”
 - SAM Local: CLI tool for local development and testing



It's demo time,
again!

- Remember the web application for this very talk we deployed earlier?

WHAT WE'RE GOING TO DO IN THIS DEMO

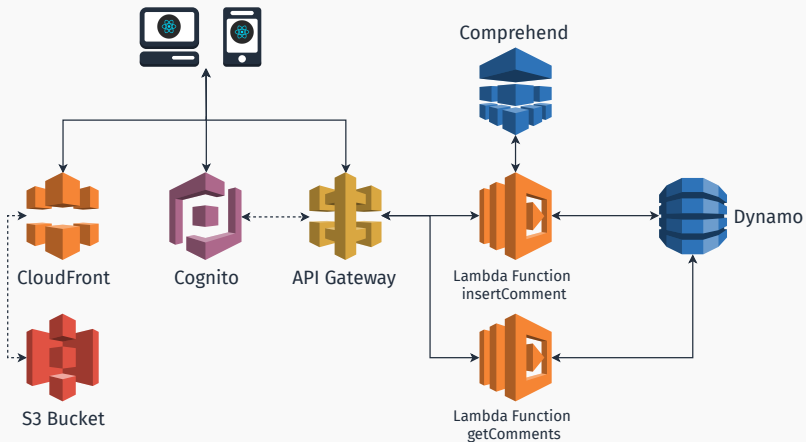
- Remember the web application for this very talk we deployed earlier?
- Now we'll make it **serverless**, and add more features:

- Remember the web application for this very talk we deployed earlier?
- Now we'll make it **serverless**, and add more features:
 - Sign-up and Authentication (Amazon Cognito)

- Remember the web application for this very talk we deployed earlier?
- Now we'll make it **serverless**, and add more features:
 - Sign-up and Authentication (Amazon Cognito)
 - Language detection and **sentiment analysis** on comments (Amazon Comprehend)

- Remember the web application for this very talk we deployed earlier?
- Now we'll make it **serverless**, and add more features:
 - Sign-up and Authentication (Amazon Cognito)
 - Language detection and **sentiment analysis** on comments (Amazon Comprehend)
 - Deploy it on a global CDN to minimize latency (Amazon CloudFront)

ARCHITECTURE



▶▶ Skip tutorial

WHAT YOU'LL NEED

- An AWS account (a free one will suffice)

WHAT YOU'LL NEED

- An AWS account (a free one will suffice)
- `git` version control (recommended)

WHAT YOU'LL NEED

- An AWS account (a free one will suffice)
- git version control (recommended)
- Node.js

WHAT YOU'LL NEED

- An AWS account (a free one will suffice)
- `git` version control (recommended)
- Node.js
- Python (recommended)

STEP 1: GET THE APP

Clone the git repository 

```
D:> git clone https://github.com/luistar/serverless-  
webapp.git serverless-webapp
```

STEP 2: INSTALL DEPENDENCIES

```
D:> cd serverless-webapp  
D:\serverless-webapp> npm install
```

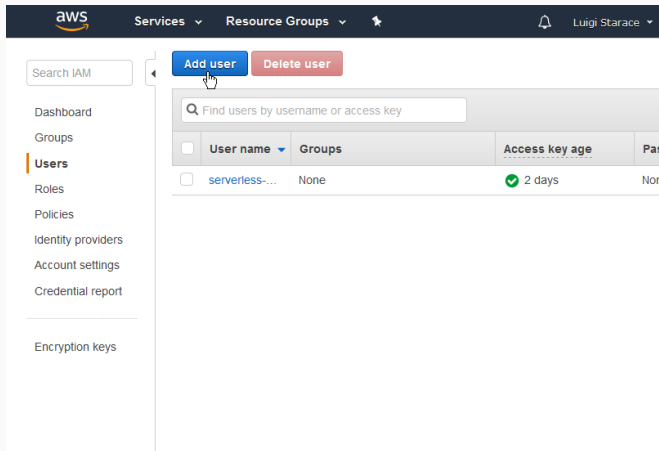
STEP 3: INSTALL AWS-MOBILE CLI TOOL

```
D:\serverless-webapp> npm -g install awsmobile-cli
```

STEP 4: CREATE A NEW USER ON IAM

In order to use `awscli` you're gonna need an access key id and a secret access key. If you don't already have one, go to the IAM console and create a new user.

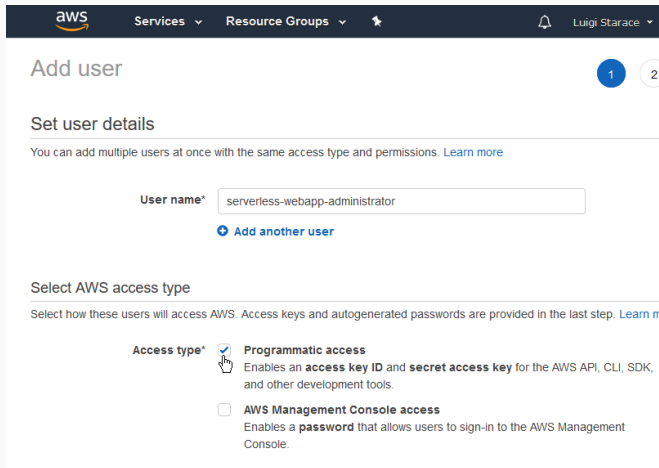
STEP 4: CREATE A NEW USER ON IAM



The screenshot shows the AWS IAM console interface. At the top, the AWS logo is on the left, and navigation links for 'Services', 'Resource Groups', and a user profile 'Luigi Starace' are on the right. A search bar labeled 'Search IAM' is positioned on the left side of the main content area. Below the search bar, a sidebar lists navigation options: 'Dashboard', 'Groups', 'Users' (highlighted with an orange bar), 'Roles', 'Policies', 'Identity providers', 'Account settings', 'Credential report', and 'Encryption keys'. In the main content area, there are two buttons: 'Add user' (blue) and 'Delete user' (red). A mouse cursor is pointing at the 'Add user' button. Below these buttons is a search input field with the placeholder text 'Find users by username or access key'. Underneath the search field is a table with the following columns: a checkbox, 'User name', 'Groups', 'Access key age', and 'Pas'. One row is visible in the table with the following data: a checkbox, the username 'serverless...', 'None' for groups, a green checkmark and '2 days' for access key age, and 'Nor' for the password status.

| <input type="checkbox"/> | User name | Groups | Access key age | Pas |
|--------------------------|---------------|--------|----------------|-----|
| <input type="checkbox"/> | serverless... | None | 2 days | Nor |

STEP 4: CREATE A NEW USER ON IAM



The screenshot shows the AWS IAM console interface. At the top, there is a navigation bar with the AWS logo, 'Services', 'Resource Groups', and a user profile for 'Luigi Starace'. The main heading is 'Add user', with a progress indicator showing step 1 of 2. Below this is the 'Set user details' section, which includes a text input field for 'User name*' containing 'serverless-webapp-administrator' and a button to 'Add another user'. The next section is 'Select AWS access type', which provides instructions and two radio button options: 'Programmatic access' (selected) and 'AWS Management Console access'.

aws Services Resource Groups Luigi Starace

Add user

Set user details

You can add multiple users at once with the same access type and permissions. [Learn more](#)

User name*

[+ Add another user](#)

Select AWS access type

Select how these users will access AWS. Access keys and autogenerated passwords are provided in the last step. [Learn more](#)

Access type* **Programmatic access**
Enables an **access key ID** and **secret access key** for the AWS API, CLI, SDK, and other development tools.

AWS Management Console access
Enables a **password** that allows users to sign-in to the AWS Management Console.

STEP 4: CREATE A NEW USER ON IAM

aws Services Resource Groups Luigi Starace

Add user

Set permissions for serverless-webapp-administrator

Attach existing policies directly

Attach one or more existing policies directly to the users or create a new policy. [Learn more](#)

Create policy Refresh

| Filter: Policy type | Search | Sho | |
|---|--------------|-------------|--------------------------------------|
| Policy name | Type | Attachments | Description |
| <input checked="" type="checkbox"/> AdministratorAccess | Job function | 1 | Provides full access to AWS services |
| <input type="checkbox"/> AlexaForBusinessDe... | AWS managed | 0 | Provide device setup access to Alex |
| <input type="checkbox"/> AlexaForBusinessFu... | AWS managed | 0 | Grants full access to AlexaForBusine |
| <input type="checkbox"/> AlexaForBusinessCa | AWS managed | 0 | Provide gateway execution access to |

STEP 4: CREATE A NEW USER ON IAM

The screenshot shows the AWS IAM console interface. At the top, the navigation bar includes the AWS logo, 'Services', 'Resource Groups', a star icon, a notification bell, and user information 'Luigi Starace', 'Global', and 'Support'. The main heading is 'Add user', with a progress indicator showing four steps: 1, 2, 3 (highlighted in blue), and 4. Below the heading is a 'Review' section with the text: 'Review your choices. After you create the user, you can view and download the autogenerated password and access key.' The 'User details' section shows: 'User name' as 'serverless-webapp-administrator' and 'AWS access type' as 'Programmatic access - with an access key'. The 'Permissions summary' section states: 'The following policies will be attached to the user shown above.' Below this is a table with two columns: 'Type' and 'Name'. The table contains one row: 'Managed policy' and 'AdministratorAccess'. At the bottom right, there are three buttons: 'Cancel', 'Previous', and 'Create user'. A mouse cursor is pointing at the 'Create user' button.

aws Services Resource Groups Luigi Starace Global Support

Add user

1 2 3 4

Review

Review your choices. After you create the user, you can view and download the autogenerated password and access key.

User details

User name serverless-webapp-administrator
AWS access type Programmatic access - with an access key

Permissions summary

The following policies will be attached to the user shown above.

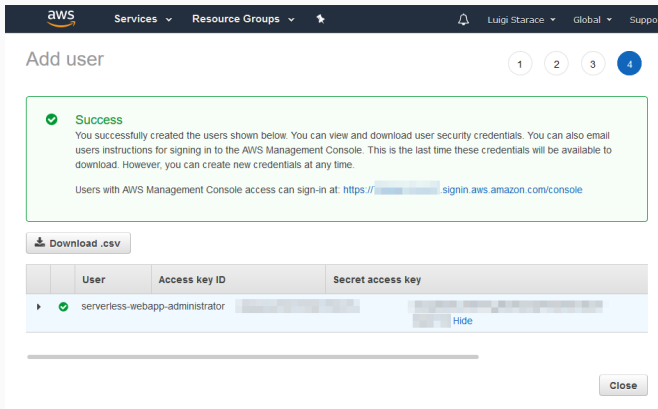
| Type | Name |
|----------------|---------------------|
| Managed policy | AdministratorAccess |

Cancel Previous Create user

STEP 4: CREATE A NEW USER ON IAM

Be sure to write up your keys and to keep them safe!

 Read more about security



The screenshot shows the AWS IAM console interface. At the top, the navigation bar includes the AWS logo, 'Services', 'Resource Groups', a search icon, a notification bell, and the user name 'Luigi Starace'. The main heading is 'Add user', with a progress indicator showing four steps, with the fourth step being active. A green success message states: 'Success. You successfully created the users shown below. You can view and download user security credentials. You can also email users instructions for signing in to the AWS Management Console. This is the last time these credentials will be available to download. However, you can create new credentials at any time. Users with AWS Management Console access can sign-in at: [https://\[redacted\].signin.aws.amazon.com/console](https://[redacted].signin.aws.amazon.com/console)'. Below the message is a 'Download .csv' button. A table displays the created user's details:

| | User | Access key ID | Secret access key |
|---|---------------------------------|---------------|-------------------|
| ▶ | serverless-webapp-administrator | [redacted] | [redacted] Hide |

A 'Close' button is located at the bottom right of the console window.

STEP 5: CONFIGURE AWSMOBILE-CLI

Configure AWS Mobile CLI.

```
D:\serverless-webapp> awsmobile configure

configure aws
? accessKeyId: <YOUR_ACCESS_KEY_ID>
? secretAccessKey: <YOUR_SECRET_ACCESS_KEY>
? region: eu-central-1
```

STEP 6: INITIALIZE A NEW AWS MOBILE PROJECT

```
D:\serverless-webapp> awsmobile init
```

```
Please tell us about your project:
```

```
? Where is your project's source directory:  src
```

```
? Where is your project's distribution directory that  
  stores build artifacts:  build
```

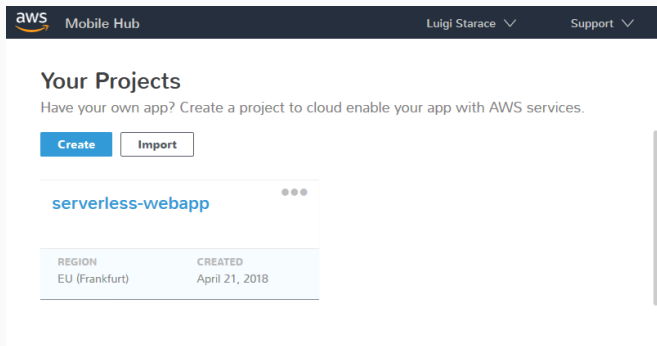
```
? What is your project's build command:  npm.cmd run-  
  script build
```

```
? What is your project's start command for local test  
  run:  npm.cmd run-script start
```

```
? What awsmobile project name would you like to use:  
  serverless-webapp
```

STEP 6: INITIALIZE A NEW AWS MOBILE PROJECT

Visit the AWS Mobile Console. Your newly created project should be waiting for you there.



The screenshot shows the AWS Mobile Hub interface. At the top, there is a dark navigation bar with the AWS logo, 'Mobile Hub', the user name 'Luigi Starace', and a 'Support' link. Below the navigation bar, the main content area is titled 'Your Projects'. It includes a sub-header 'Have your own app? Create a project to cloud enable your app with AWS services.' and two buttons: 'Create' (highlighted in blue) and 'Import'. Below the buttons, there is a card for a project named 'serverless-webapp'. The card displays the project name, a three-dot menu icon, and a table with the following data:

| REGION | CREATED |
|----------------|----------------|
| EU (Frankfurt) | April 21, 2018 |

Starter Kits and Tutorials

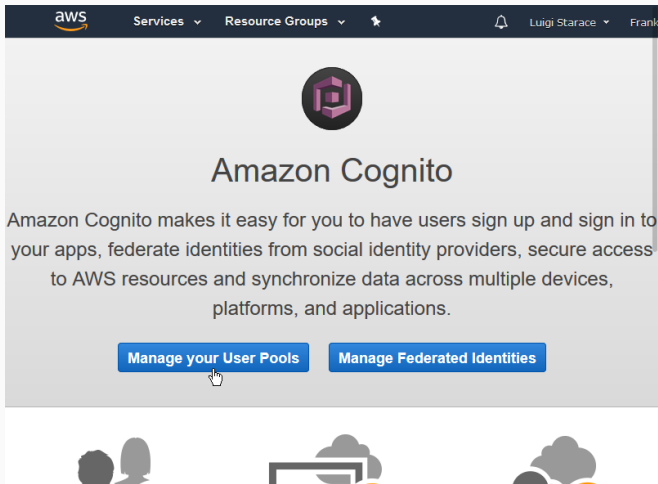
No app? Kick the tires with one of our cloud enabled starter kits. Or follow a step-by-step tutorial to cloud enable a sample app yourself.

STEP 7: CREATE A COGNITO USER POOL

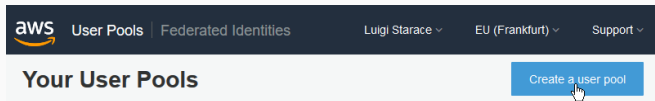
First we're gonna need a Cognito User Pool to authenticate our users. Let's create one.

STEP 7: CREATE A COGNITO USER POOL

Visit the Cognito Console.



STEP 7: CREATE A COGNITO USER POOL



The screenshot shows the AWS IAM console interface. At the top, there is a dark navigation bar with the AWS logo on the left, followed by the text 'User Pools | Federated Identities'. On the right side of this bar are three dropdown menus: 'Luigi Starace', 'EU (Frankfurt)', and 'Support'. Below the navigation bar, the main content area has a light gray background. On the left, the text 'Your User Pools' is displayed in a large, bold font. On the right side of this area, there is a blue button with the text 'Create a user pool'. A mouse cursor is positioned over the button, indicating it is being clicked.

aws User Pools | Federated Identities Luigi Starace ▾ EU (Frankfurt) ▾ Support ▾

Your User Pools [Create a user pool](#)

STEP 7: CREATE A COGNITO USER POOL

Insert a name for your user pool.

What do you want to name your user pool?

Give your user pool a descriptive name so you can easily identify it in the future.

Pool name

How do you want to create your user pool?

Review defaults

Start by reviewing the defaults and then customize as desired

Step through settings

Step through each setting to make your choices

STEP 7: CREATE A COGNITO USER POOL

Make sure only an email is required.

Which standard attributes do you want to require?

All of the standard attributes can be used for user profiles, but the attributes you select will be required for sign up. You will not be able to change these requirements after the pool is created. If you select an attribute to be an alias, users will be able to sign-in using that value or their username. [Learn more about attributes.](#)

| Required | Attribute | Required | Attribute |
|-------------------------------------|-------------|--------------------------|--------------------|
| <input type="checkbox"/> | address | <input type="checkbox"/> | nickname |
| <input type="checkbox"/> | birthdate | <input type="checkbox"/> | phone number |
| <input checked="" type="checkbox"/> | email | <input type="checkbox"/> | picture |
| <input type="checkbox"/> | family name | <input type="checkbox"/> | preferred username |
| <input type="checkbox"/> | gender | <input type="checkbox"/> | profile |
| <input type="checkbox"/> | given name | <input type="checkbox"/> | zoneinfo |
| <input type="checkbox"/> | locale | <input type="checkbox"/> | updated at |
| <input type="checkbox"/> | middle name | <input type="checkbox"/> | website |
| <input type="checkbox"/> | name | | |

STEP 7: CREATE A COGNITO USER POOL

Review your configuration and create the pool.

aws User Pools Federated Identities Luigi Starace - EU (Frankfurt) - Support -

Create a user pool Cancel

Name

Attributes

Policies

MFA and verifications

Message customizations

Tags

Devices

App clients

Triggers

Review

Pool name serverless-webapp-user-pool

Required attributes email

Alias attributes [Choose alias attributes...](#)

Username attributes [Choose username attributes...](#)

Custom attributes [Choose custom attributes...](#)

Minimum password length 8

Password policy no requirements

User sign ups allowed? Users can sign themselves up

MFA [Enable MFA...](#)

Verifications Email

Tags [Choose tags for your user pool](#)

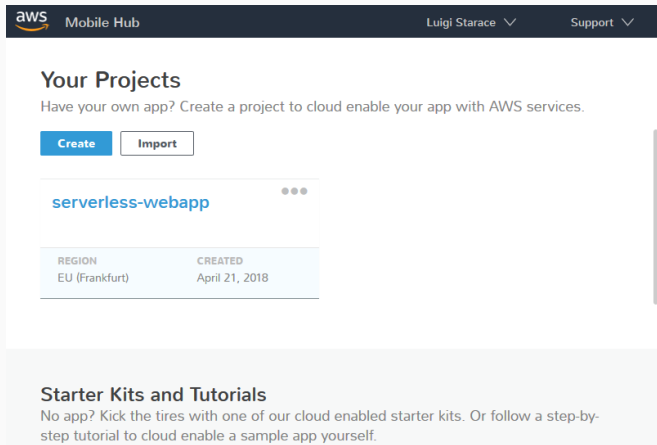
App clients [Add app client...](#)

Triggers [Add triggers...](#)

[Create pool](#)

STEP 8: CONFIGURE USER SIGN-IN IN THE MOBILE APP CONSOLE

Return to the AWS Mobile Console and open your project.



The screenshot shows the AWS Mobile Hub interface. At the top, there is a navigation bar with the AWS logo, 'Mobile Hub', the user name 'Luigi Starace', and a 'Support' link. Below the navigation bar, the main heading is 'Your Projects'. Underneath this heading is a sub-heading: 'Have your own app? Create a project to cloud enable your app with AWS services.' There are two buttons: a blue 'Create' button and a white 'Import' button. Below the buttons is a card for a project named 'serverless-webapp'. To the right of the project name are three dots. Below the project name is a table with two columns: 'REGION' and 'CREATED'. The table contains one row with the values 'EU (Frankfurt)' and 'April 21, 2018'.

Your Projects

Have your own app? Create a project to cloud enable your app with AWS services.

[Create](#) [Import](#)

serverless-webapp

| REGION | CREATED |
|----------------|----------------|
| EU (Frankfurt) | April 21, 2018 |

Starter Kits and Tutorials

No app? Kick the tires with one of our cloud enabled starter kits. Or follow a step-by-step tutorial to cloud enable a sample app yourself.

STEP 8: CONFIGURE USER SIGN-IN IN THE MOBILE APP CONSOLE

Add user sign in to the project.

Add More Backend Features



User Sign-in

Let your users sign in with public identity providers or your own identity system.

Powered by Amazon Cognito



NoSQL Database

Store data in a fully managed cloud database.

Powered by Amazon DynamoDB



User File Storage

Store files in the cloud.

Powered by Amazon S3



Conversational Bots

Add voice and chat bots to your mobile app.

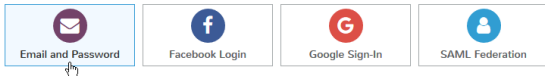
Powered by Amazon Lex



STEP 8: CONFIGURE USER SIGN-IN IN THE MOBILE APP CONSOLE

Import your newly created user pool.

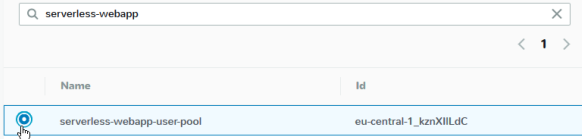
Add sign-in Providers



Create new or import



Select user pool



STEP 8: CONFIGURE USER SIGN-IN IN THE MOBILE APP CONSOLE

Pull your new project configuration with

```
D:\serverless-webapp> awsmobile pull
```

If you were to start the application locally with

```
D:\serverless-webapp> npm start
```

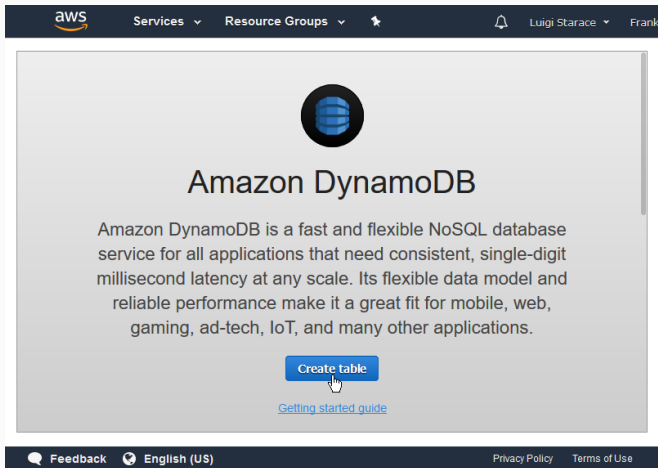
The authentication will now work.

STEP 9: CONFIGURE AMAZON DYNAMO DATABASE

Next thing we're gonna need is a database to store the comments. In this tutorial we'll use the NoSQL database Amazon Dynamo.

STEP 9: CONFIGURE AMAZON DYNAMO DATABASE

Visit the Dynamo Dashboard.



The screenshot shows the AWS Management Console interface for the Amazon DynamoDB dashboard. At the top, the AWS logo is on the left, and navigation options for 'Services', 'Resource Groups', and user information ('Luigi Starace') are on the right. The main content area features the DynamoDB logo (a blue globe with horizontal lines) and the title 'Amazon DynamoDB'. Below the title is a descriptive paragraph: 'Amazon DynamoDB is a fast and flexible NoSQL database service for all applications that need consistent, single-digit millisecond latency at any scale. Its flexible data model and reliable performance make it a great fit for mobile, web, gaming, ad-tech, IoT, and many other applications.' A prominent blue button labeled 'Create table' is centered below the text, with a mouse cursor hovering over it. Below the button is a blue link labeled 'Getting started guide'. The footer of the console contains 'Feedback', 'English (US)', 'Privacy Policy', and 'Terms of Use'.

STEP 9: CONFIGURE AMAZON DYNAMO DATABASE

Create a new Comments table as shown. Leave other fields with their default values.

The screenshot shows the AWS Management Console interface for creating a new DynamoDB table. At the top, the AWS logo and navigation menu are visible. The main heading is "Create DynamoDB table" with a "Tutorial" button. Below the heading, there is a brief description of DynamoDB. The configuration section includes:

- Table name***: A text input field containing "Comments".
- Primary key***: A section for defining the primary key, including:
 - Partition key**: A text input field containing "User", with a dropdown menu set to "String".
 - Add sort key**: A checkbox that is checked.
 - Sort key**: A text input field containing "Timestamp", with a dropdown menu set to "String".

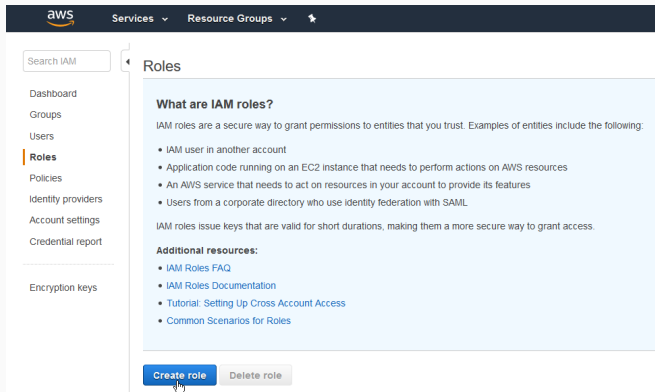
Below the configuration section, the "Table settings" section is partially visible, with the text: "Default settings provide the fastest way to get started with your table. You can modify these default settings now or after your table has been created."

STEP 10: CREATE A ROLE FOR THE LAMBDA FUNCTIONS

Before we create our Lambda functions, let's create a new role defining the authorizations we want them to have.

STEP 10: CREATE A ROLE FOR THE LAMBDA FUNCTIONS

Return to the IAM console and select the “role” tab, then the “create role” button.



The screenshot shows the AWS IAM console interface. At the top, there is a navigation bar with the AWS logo, 'Services', and 'Resource Groups'. On the left, a sidebar contains a search box labeled 'Search IAM' and a list of navigation items: Dashboard, Groups, Users, Roles (highlighted with an orange bar), Policies, Identity providers, Account settings, Credential report, and Encryption keys. The main content area is titled 'Roles' and features a light blue informational box. This box contains the heading 'What are IAM roles?', a paragraph explaining that IAM roles are a secure way to grant permissions, and a bulleted list of examples: IAM user in another account, application code on an EC2 instance, an AWS service, and users from a corporate directory. Below this, it states that IAM roles issue keys for short durations. Further down, there is a section for 'Additional resources' with links to 'IAM Roles FAQ', 'IAM Roles Documentation', 'Tutorial: Setting Up Cross Account Access', and 'Common Scenarios for Roles'. At the bottom of the main content area, there are two buttons: 'Create role' (highlighted in blue) and 'Delete role'.

STEP 10: CREATE A ROLE FOR THE LAMBDA FUNCTIONS

Select “AWS Service” and “Lambda” in the wizard, as shown in the picture.

The screenshot shows the AWS IAM console interface for creating a role. At the top, the AWS logo and navigation menus are visible. The main heading is 'Create role'. The first step is 'Select type of trusted entity', with four options: 'AWS service' (selected), 'Another AWS account', 'Web identity', and 'SAML'. Below this, the second step is 'Choose the service that will use this role', with a list of services including 'EC2', 'Lambda' (selected), 'API Gateway', 'AppSync', 'Application Auto Scaling', 'Config', 'DMS', 'Data Pipeline', 'Elastic Container Service', 'Elastic Transcoder', 'ElasticLoadBalancing', 'Lex', 'Machine Learning', and 'MediaConvert'. At the bottom right, there are 'Cancel' and 'Next: Per' buttons.

aws Services ▾ Resource Groups ▾ Luigi Starace ▾

Create role

Select type of trusted entity

- AWS service**
EC2, Lambda and others
- Another AWS account
Belonging to you or 3rd party
- Web identity
Cognito or any OpenID provider
- SAML

Allows AWS services to perform actions on your behalf. [Learn more](#)

Choose the service that will use this role

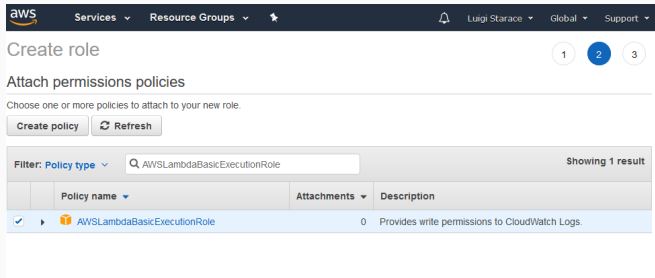
- EC2**
Allows EC2 instances to call AWS services on your behalf.
- Lambda**
Allows Lambda functions to call AWS services on your behalf.

| | | | |
|--------------------------|---------------|---------------------------|------------------|
| API Gateway | Config | Elastic Container Service | Lex |
| AppSync | DMS | Elastic Transcoder | Machine Learning |
| Application Auto Scaling | Data Pipeline | ElasticLoadBalancing | MediaConvert |


Cancel Next: Per

STEP 10: CREATE A ROLE FOR THE LAMBDA FUNCTIONS

Add the AWSLambdaBasicExecutionRole, as shown.

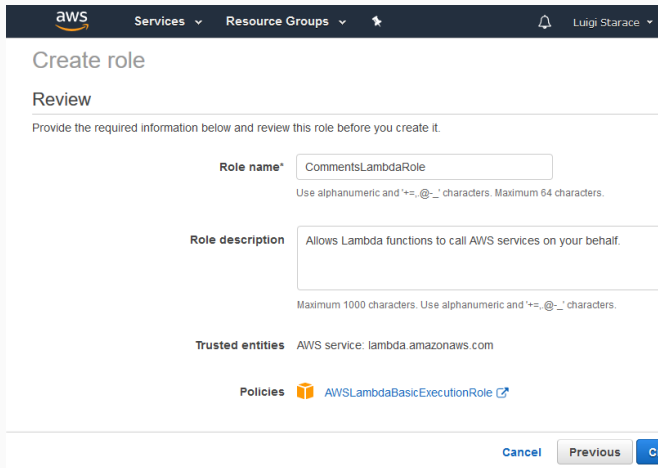


The screenshot shows the AWS IAM console interface. At the top, the navigation bar includes the AWS logo, 'Services', 'Resource Groups', a search icon, a notification bell, the user name 'Luigi Starace', 'Global', and 'Support'. The main heading is 'Create role', with three numbered steps (1, 2, 3) and step 2 is highlighted. Below the heading is the section 'Attach permissions policies' with the instruction 'Choose one or more policies to attach to your new role.' and buttons for 'Create policy' and 'Refresh'. A search bar contains the text 'AWSLambdaBasicExecutionRole' and shows 'Showing 1 result'. Below the search bar is a table with columns for 'Policy name', 'Attachments', and 'Description'. One row is visible, representing the 'AWSLambdaBasicExecutionRole' policy, which has 0 attachments and a description of 'Provides write permissions to CloudWatch Logs.'.

| | Policy name | Attachments | Description |
|-------------------------------------|---|-------------|--|
| <input checked="" type="checkbox"/> |  AWSLambdaBasicExecutionRole | 0 | Provides write permissions to CloudWatch Logs. |

STEP 10: CREATE A ROLE FOR THE LAMBDA FUNCTIONS

Insert a name and a description and create the role.



The screenshot shows the AWS IAM console interface for creating a new role. At the top, the AWS logo is on the left, and navigation options for 'Services', 'Resource Groups', and a user profile 'Luigi Starace' are on the right. The main heading is 'Create role', followed by a sub-heading 'Review'. Below this, a note states: 'Provide the required information below and review this role before you create it.'

The 'Role name*' field contains the text 'CommentsLambdaRole'. A tooltip below it reads: 'Use alphanumeric and '+=, @, _' characters. Maximum 64 characters.'

The 'Role description' field contains the text 'Allows Lambda functions to call AWS services on your behalf.' A tooltip below it reads: 'Maximum 1000 characters. Use alphanumeric and '+=, @, _' characters.'

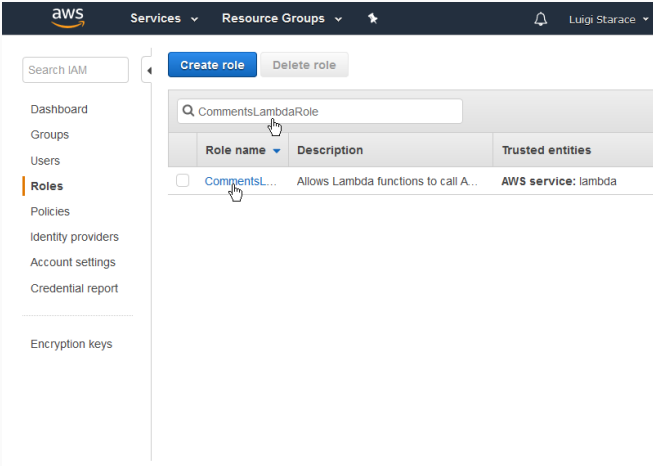
The 'Trusted entities' section shows 'AWS service: lambda.amazonaws.com'.

The 'Policies' section shows a single policy: 'AWSLambdaBasicExecutionRole' with an external link icon.

At the bottom right, there are three buttons: 'Cancel' (text), 'Previous' (disabled), and 'Create' (partially visible).

STEP 10: CREATE A ROLE FOR THE LAMBDA FUNCTIONS

Go back to the roles tab in the IAM Dashboard and select your newly created role.



The screenshot shows the AWS IAM console interface. At the top, there is a navigation bar with the AWS logo, 'Services', 'Resource Groups', and a user profile 'Luigi Starace'. On the left, a sidebar contains navigation links: 'Search IAM', 'Dashboard', 'Groups', 'Users', 'Roles' (highlighted with an orange bar), 'Policies', 'Identity providers', 'Account settings', 'Credential report', and 'Encryption keys'. The main content area has two buttons: 'Create role' (blue) and 'Delete role' (grey). Below these is a search bar containing 'CommentsLambdaRole'. A table lists the roles, with one role visible:

| Role name | Description | Trusted entities |
|---|--------------------------------------|----------------------------|
| <input type="checkbox"/> CommentsL... | Allows Lambda functions to call A... | AWS service: lambda |

STEP 10: CREATE A ROLE FOR THE LAMBDA FUNCTIONS

Add inline policies to allow the role to access Dynamo DB and Comprehend.

The screenshot shows the AWS IAM console interface for an IAM role. The top navigation bar includes 'Resources', 'Resource Groups', a search icon, a notification bell, and user information 'Luigi Starace'. The main content area is titled 'Summary' and includes a 'Delete role' button. The role details are as follows:

- Role ARN:** `arn:aws:iam::788880174327:role/CommentsLambdaRole`
- Role description:** Allows Lambda functions to call AWS services on your behalf. | [Edit](#)
- Instance Profile ARNs:** [Copy](#)
- Path:** /
- Creation time:** 2018-04-21 12:19 UTC+0200
- Maximum CLI/API session duration:** 1 hour [Edit](#)

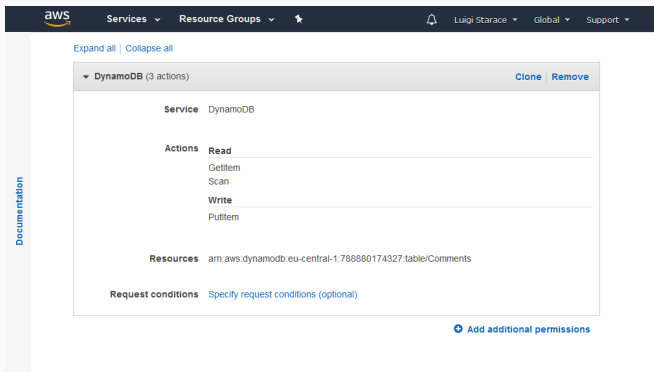
Below the summary, there are four tabs: 'Permissions', 'Trust relationships', 'Access Advisor', and 'Revoke sessions'. The 'Permissions' tab is active, showing an 'Attach policy' button and 'Attached policies: 1'. A table lists the attached policy:

| Policy name | Policy type |
|---|--------------------|
| AWSLambdaBasicExecutionRole | AWS managed policy |

At the bottom right of the 'Permissions' section, there is a blue button with a plus icon and the text 'Add inline policy', which has a hand cursor hovering over it.

STEP 10: CREATE A ROLE FOR THE LAMBDA FUNCTIONS

Add an inline policy to allow this role to access Dynamo DB tables.



The screenshot shows the AWS IAM console interface. At the top, there is a navigation bar with the AWS logo, 'Services', 'Resource Groups', a search icon, a notification bell, and user information 'Luigi Starace', 'Global', and 'Support'. Below the navigation bar, there are links for 'Expand all' and 'Collapse all'. The main content area displays a policy configuration for 'DynamoDB (3 actions)'. On the left side of the console, there is a vertical label 'Documentation'. The policy configuration includes:

- Service:** DynamoDB
- Actions:**
 - Read:** GetItem, Scan
 - Write:** PutItem
- Resources:** arn:aws:dynamodb:eu-central-1:788880174327:table/Comments
- Request conditions:** Specify request conditions (optional)

At the bottom right of the configuration box, there is a button labeled 'Add additional permissions' with a plus icon.

STEP 10: CREATE A ROLE FOR THE LAMBDA FUNCTIONS

Save the inline policy.

The screenshot shows the AWS IAM console interface for creating a policy. The top navigation bar includes the AWS logo, 'Services', 'Resource Groups', a star icon, a notification bell, 'Luigi Starace', 'Global', and 'Support'. The main heading is 'Create policy' with two numbered steps: step 1 is inactive and step 2 is active. Below the heading is the 'Review policy' section, which includes a sub-heading and a note: 'Before you create this policy, provide the required information and review this policy.' The 'Name*' field contains 'DynamoCommentsPolicy' with a note below it: 'Maximum 128 characters. Use alphanumeric and '+-,@_.' characters.' The 'Summary' section features a search filter and a table of permissions.

| Service | Access level | Resource |
|--|----------------------|-------------------------------|
| Allow (1 of 136 services) Show remaining 135 | | |
| DynamoDB | Limited: Read, Write | TableName string like Com |

STEP 10: CREATE A ROLE FOR THE LAMBDA FUNCTIONS

Now add another inline policy to allow this role to access Comprehend's detect language and detect sentiment features.

The screenshot shows the AWS IAM console interface for creating a new policy. At the top, the navigation bar includes the AWS logo, 'Services', 'Resource Groups', and user information for 'Luigi Starace'. The main heading is 'Create policy', with two numbered steps (1 and 2) in the top right corner. Below the heading is a descriptive paragraph: 'A policy defines the AWS permissions that you can assign to a user, group, or role. You can create and edit a policy in the visual editor and using JSON. [Learn more](#)'. There are two tabs: 'Visual editor' (selected) and 'JSON'. An 'Import managed policy' link is on the right. Below the tabs are 'Expand all' and 'Collapse all' links. A section titled 'Comprehend (2 actions)' is expanded, showing 'Service: Comprehend', 'Actions: Read' (with sub-items 'DetectDominantLanguage' and 'DetectSentiment'), 'Resources: All resources have been selected for you because this service does not allow you to choose specific resources.', and 'Request conditions: Specify request conditions (optional)'. At the bottom right of this section are 'Clone' and 'Remove' links. A blue plus icon and the text 'Add additional permissions' are at the very bottom.

STEP 10: CREATE A ROLE FOR THE LAMBDA FUNCTIONS

Your role should look like this.

The screenshot shows the AWS IAM console interface for a role named 'CommentsLambdaRole'. The breadcrumb navigation is 'Roles > CommentsLambdaRole'. A 'Delete role' button is visible in the top right corner.

Summary

- Role ARN:** `arn:aws:iam::788880174327:role/CommentsLambdaRole`
- Role description:** Allows Lambda functions to call AWS services on your behalf. | [Edit](#)
- Instance Profile ARNs:** [Copy](#)
- Path:** /
- Creation time:** 2018-04-21 12:19 UTC+0200
- Maximum CLI/API session duration:** 1 hour [Edit](#)

Permissions | Trust relationships | Access Advisor | Revoke sessions

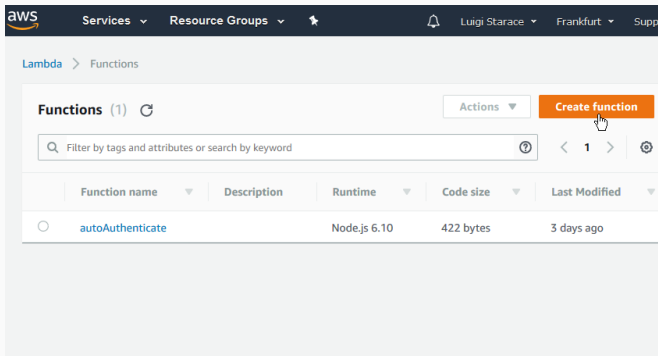
[Attach policy](#) Attached policies: 3

| Policy name | Policy type | |
|-----------------------------|--------------------|-------------------|
| AWSLambdaBasicExecutionRole | AWS managed policy | ✕ |
| ComprehendCommentsPolicy | Inline policy | ✕ |
| DynamoCommentsPolicy | Inline policy | ✕ |

[+ Add inline policy](#)

STEP 11: CREATE THE LAMBDA FUNCTIONS

Go to the Lambda Dashboard and click on the “Create function” button.



The screenshot shows the AWS Lambda console interface. At the top, the navigation bar includes the AWS logo, 'Services', 'Resource Groups', a search icon, a notification bell, and the user's name 'Luigi Starace' with a dropdown arrow, the region 'Frankfurt', and 'Supp'. Below the navigation bar, the breadcrumb 'Lambda > Functions' is visible. The main content area features a header 'Functions (1)' with a refresh icon and an 'Actions' dropdown menu. A prominent orange 'Create function' button is located to the right of the 'Actions' menu, with a mouse cursor hovering over it. Below the header is a search bar with the placeholder text 'Filter by tags and attributes or search by keyword'. A pagination control shows '< 1 >' and a settings icon. A table below displays the details of the existing function:

| | Function name | Description | Runtime | Code size | Last Modified |
|-----------------------|------------------|-------------|--------------|-----------|---------------|
| <input type="radio"/> | autoAuthenticate | | Node.js 6.10 | 422 bytes | 3 days ago |

STEP 11: CREATE THE LAMBDA FUNCTIONS

Select “Author from scratch”

The screenshot shows the AWS Lambda console interface for creating a new function. The navigation bar at the top includes the AWS logo, 'Services', 'Resource Groups', and user information. The breadcrumb trail indicates the path: Lambda > Functions > Create function. The main heading is 'Create function'. Three options are presented as cards:

- Author from scratch** (selected): Start with a simple "hello world" example. Includes an icon of a document and a gear.
- Blueprints**: Choose a preconfigured template as a starting point for your Lambda function. Includes an icon of three documents with checkmarks.
- Serverless Application Repository**: Find and deploy serverless applications published by development companies, and partners. Includes an icon of a cloud and a document.

Below the options, the 'Author from scratch' section is expanded, showing a sub-heading 'Author from scratch Info' and a 'Name' field with the value 'insertComment'. A 'Runtime' field is partially visible below.

STEP 11: CREATE THE LAMBDA FUNCTIONS

Name the function `insertComment` and select Node.js 6.10 as the runtime and the role we created earlier as the role.

Author from scratch [Info](#)

Name

Runtime

Role
Defines the permissions of your function. Note that new roles may not be available for a few minutes after creation. [Learn more](#) about Lambda execution roles.

Existing role
You may use an existing role with this function. Note that the role must be assumable by Lambda and must have Cloudwatch Logs permissions.

[Cancel](#) [Create function](#)

STEP 11: CREATE THE LAMBDA FUNCTIONS

Insert the code provided in the `lambda/insertComment.js` file in the next screen, then save the lambda function.

STEP 11: CREATE THE LAMBDA FUNCTIONS

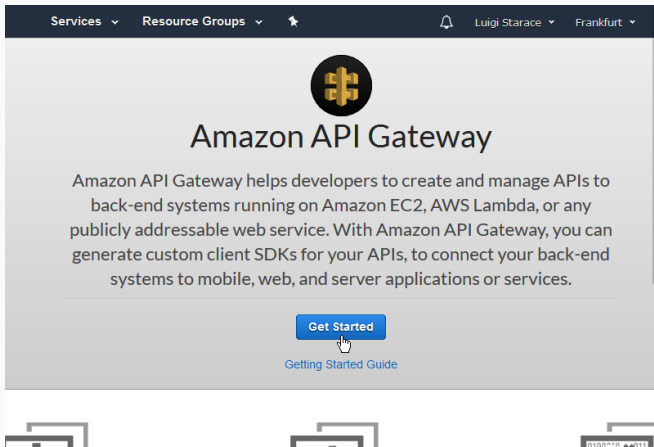
Proceed similarly and create the `getComments` Lambda function.

STEP 12: CREATE THE APIS TO EXPOSE THE LAMBDA FUNCTIONS

Once we have our Lambda functions, let's hook 'em up with an API our web app can rely upon.

STEP 12: CREATE THE APIS TO EXPOSE THE LAMBDA FUNCTIONS

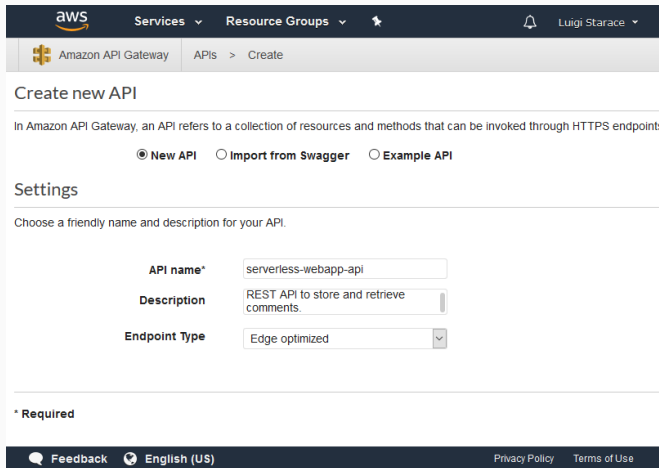
Visit the API Gateway Dashboard



The screenshot shows the Amazon API Gateway dashboard within the AWS Management Console. At the top, there is a dark navigation bar with 'Services' and 'Resource Groups' dropdown menus, a home icon, a notification bell, and the user's name 'Luigi Starace' with a location dropdown set to 'Frankfurt'. The main content area features the API Gateway logo (a gold double-dollar sign inside a black circle) and the heading 'Amazon API Gateway'. Below this, a paragraph describes the service: 'Amazon API Gateway helps developers to create and manage APIs to back-end systems running on Amazon EC2, AWS Lambda, or any publicly addressable web service. With Amazon API Gateway, you can generate custom client SDKs for your APIs, to connect your back-end systems to mobile, web, and server applications or services.' A prominent blue 'Get Started' button is centered, with a mouse cursor hovering over it. Below the button is a link for the 'Getting Started Guide'. At the bottom of the dashboard, there are three placeholder icons for API resources.

STEP 12: CREATE THE APIS TO EXPOSE THE LAMBDA FUNCTIONS

Create a new API and select a name and a description.



The screenshot shows the AWS console interface for creating a new API. At the top, the AWS logo is on the left, and navigation links for 'Services', 'Resource Groups', and a user profile 'Luigi Starace' are on the right. Below the navigation bar, the breadcrumb path is 'Amazon API Gateway > APIs > Create'. The main heading is 'Create new API'. A sub-heading explains that an API in Amazon API Gateway is a collection of resources and methods that can be invoked through HTTPS endpoints. There are three radio button options: 'New API' (selected), 'Import from Swagger', and 'Example API'. Below this is a 'Settings' section with the instruction 'Choose a friendly name and description for your API.' There are three form fields: 'API name*' with the value 'serverless-webapp-api', 'Description' with the value 'REST API to store and retrieve comments.', and 'Endpoint Type' with a dropdown menu set to 'Edge optimized'. At the bottom left, there is a '* Required' label. The footer contains 'Feedback', 'English (US)', 'Privacy Policy', and 'Terms of Use'.

aws Services Resource Groups Luigi Starace

Amazon API Gateway APIs > Create

Create new API

In Amazon API Gateway, an API refers to a collection of resources and methods that can be invoked through HTTPS endpoints

New API Import from Swagger Example API

Settings

Choose a friendly name and description for your API.

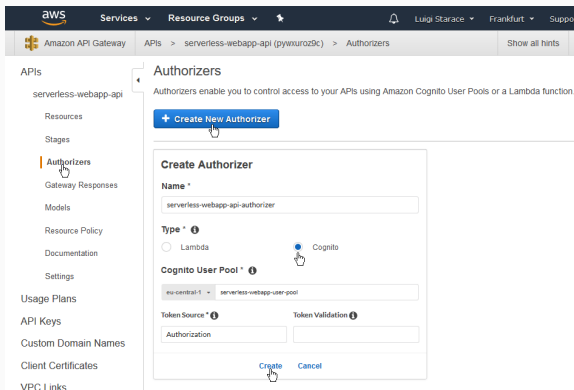
| | |
|---------------|--|
| API name* | serverless-webapp-api |
| Description | REST API to store and retrieve comments. |
| Endpoint Type | Edge optimized |

* Required

Feedback English (US) Privacy Policy Terms of Use

STEP 12: CREATE THE APIS TO EXPOSE THE LAMBDA FUNCTIONS

Select the Authorizers tab and create a new Authorizer for your API. Give it a name, select the user pool we created earlier, and enter “Authorization” in the “Token Source” field.



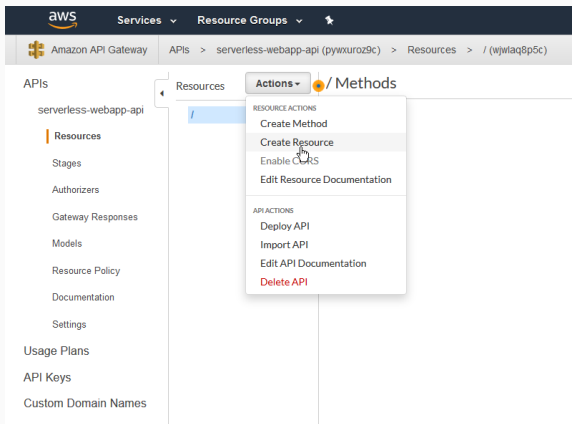
The screenshot displays the AWS Management Console interface for creating a new Authorizer. The breadcrumb navigation shows the path: Amazon API Gateway > APIs > serverless-webapp-api (pywuroz9c) > Authorizers. A sidebar on the left lists various API Gateway components, with 'Authorizers' selected. The main content area is titled 'Authorizers' and includes a '+ Create New Authorizer' button. Below this is the 'Create Authorizer' form with the following fields:

- Name ***: serverless-webapp-api-authorizer
- Type ***: Lambda, Cognito
- Cognito User Pool ***: eu-central-1 - serverless-webapp-user-pool
- Token Source ***: Authorization
- Token Validation ***: (empty)

At the bottom of the form are 'Create' and 'Cancel' buttons.

STEP 12: CREATE THE APIS TO EXPOSE THE LAMBDA FUNCTIONS


Select the resources tab and create a new Resource




STEP 12: CREATE THE APIS TO EXPOSE THE LAMBDA FUNCTIONS

Name the resource comments, enable CORS and continue.

New Child Resource

Use this page to create a new child resource for your resource. 

Configure as [proxy resource](#) 

Resource Name*

Resource Path*

You can add path parameters using brackets. For example, the resource path **{username}** represents a path parameter called 'username'. Configuring **/{proxy+}** as a proxy resource catches all requests to its sub-resources. For example, it works for a GET request to /foo. To handle requests to /, add a new ANY method on the / resource.

Enable API Gateway CORS 

* Required [Cancel](#) [Create Resource](#)

STEP 12: CREATE THE APIS TO EXPOSE THE LAMBDA FUNCTIONS

Select the comments resource and create a new POST method.

The screenshot displays the AWS Management Console interface for an Amazon API Gateway. The breadcrumb navigation at the top indicates the path: Services > Resource Groups > Amazon API Gateway > APIs > serverless-webapp-api (pywxuroz9c) > Resources > /comments (hnh412). On the left sidebar, the 'Resources' section is expanded, showing a tree view with a folder icon for '/comments'. The main content area shows the 'Actions' dropdown menu for the selected resource, with 'Create Resource' highlighted. The menu is divided into 'RESOURCE ACTIONS' and 'API ACTIONS'. The 'RESOURCE ACTIONS' include 'Create Method', 'Create Resource', 'Enable CORS', 'Edit Resource Documentation', and 'Delete Resource'. The 'API ACTIONS' include 'Deploy API', 'Import API', 'Edit API Documentation', and 'Delete API'. The background shows the 'Methods' tab for the resource, with a table containing one entry with a status of 'None' and 'Not required'.

STEP 12: CREATE THE APIS TO EXPOSE THE LAMBDA FUNCTIONS

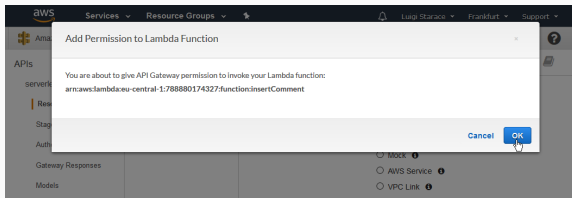
As shown, select the insertComment Lambda function you created earlier as the integration point.

The screenshot shows the AWS API Gateway console configuration for a new method. The breadcrumb path is Resources > Actions > /comments - POST - Setup. The left sidebar shows the resource tree with /comments > OPTIONS > POST selected. The main panel contains the following configuration:

- Integration type:** Radio buttons for Lambda Function (selected), HTTP, Mock, AWS Service, and VPC Link.
- Use Lambda Proxy Integration:** Checked checkbox.
- Lambda Region:** Dropdown menu set to eu-central-1.
- Lambda Function:** Text input field containing 'insertComment'.
- Use Default Timeout:** Checked checkbox.
- Save:** A blue button at the bottom right.

STEP 12: CREATE THE APIS TO EXPOSE THE LAMBDA FUNCTIONS

Give API Gateway the permission to invoke the Lambda function



STEP 12: CREATE THE APIS TO EXPOSE THE LAMBDA FUNCTIONS

Select the POST method on the comments resource, then select the Method Request card.

The screenshot displays the AWS API Gateway console interface for configuring a POST method on the /comments resource. The breadcrumb navigation shows the path: APIs > serverless-webapp-api (pywuroz9c) > Resources > /comments (hnh412) > POST. The left-hand navigation pane shows the resource tree with /comments expanded and the POST method selected. The main content area is titled "/comments - POST - Method Execution" and features a flow diagram with four components:

- Method Request:** Contains configuration for the request, including **Auth:** NONE and **ARN:** arn:aws:execute-api:eu-central-1:788880174327:pywuroz9c*/POST/comments.
- Integration Request:** Shows the **Type:** LAMBDA_PROXY.
- Integration Response:** Includes a note: "Proxy integrations cannot be configured to transform responses."
- Method Response:** Shows **HTTP Status:** Proxy and **Models:** application/json => Empty.

Vertical bars on the left and right sides of the diagram are labeled "Client" and "Lambda InsertComment" respectively, with arrows indicating the flow of data between the components.

STEP 12: CREATE THE APIS TO EXPOSE THE LAMBDA FUNCTIONS

Select the authorizer you created earlier for the Authorization field in the Settings section.

The screenshot shows the AWS IAM console interface for configuring a REST API method. The breadcrumb navigation is: APIs > serverless-webapp-api (pywxuroz9c) > Resources > /comments (hnh412) > POST. The page title is "/comments - POST - Method Request".

On the left, a tree view shows the resource structure: / > /comments > OPTIONS > POST (selected).

The main content area is titled "Settings" and contains the following configuration options:

- Authorization:** A dropdown menu is set to "serverless-webapp-api-authorizer". An orange arrow points to this dropdown.
- Request Validator:** Set to "NONE".
- API Key Required:** Set to "false".

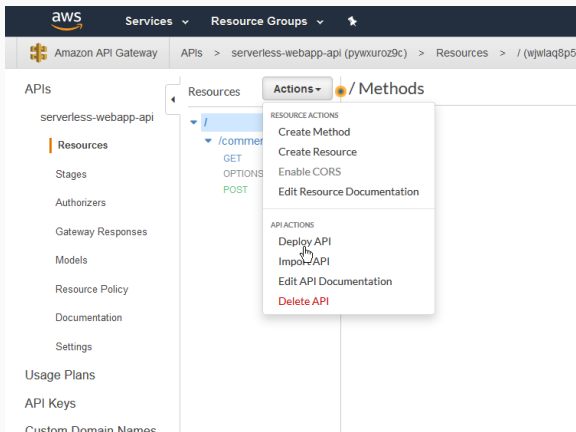
Below these settings are expandable sections for "URL Query String Parameters", "HTTP Request Headers", "Request Body", and "SDK Settings".

STEP 12: CREATE THE APIS TO EXPOSE THE LAMBDA FUNCTIONS

Proceed similarly to hook up the GET method with the `getComments` Lambda function. This time authorization is not needed. We want non-authenticated users to be able to fetch the comments.

STEP 12: CREATE THE APIS TO EXPOSE THE LAMBDA FUNCTIONS

Once you are done setting up the GET method, select the root resource, then Deploy API.

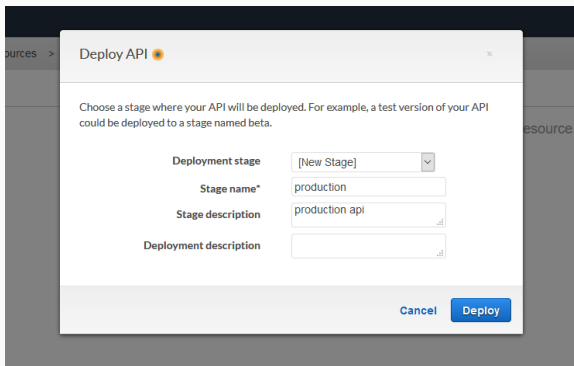


The screenshot displays the AWS Management Console interface for an Amazon API Gateway. The breadcrumb navigation shows the path: Amazon API Gateway > APIs > serverless-webapp-api (pywxuroz9c) > Resources > / (wjwlaq8p5). The left-hand navigation pane lists various API Gateway components, with 'Resources' selected. The main content area shows a tree view of resources under the root path '/'. The root resource is selected, and its 'Actions' menu is open. The menu is divided into two sections: 'RESOURCE ACTIONS' and 'API ACTIONS'. The 'API ACTIONS' section includes the 'Deploy API' option, which is highlighted by a mouse cursor. Other options in the 'API ACTIONS' section include 'Import API', 'Edit API Documentation', and 'Delete API'.

- RESOURCE ACTIONS
 - Create Method
 - Create Resource
 - Enable CORS
 - Edit Resource Documentation
- API ACTIONS
 - Deploy API
 - Import API
 - Edit API Documentation
 - Delete API

STEP 12: CREATE THE APIS TO EXPOSE THE LAMBDA FUNCTIONS

Insert deployment stage informations and deploy.



The screenshot shows a 'Deploy API' dialog box with the following fields and options:

- Deployment stage:** A dropdown menu currently showing '[New Stage]'.
- Stage name*:** A text input field containing 'production'.
- Stage description:** A text input field containing 'production api'.
- Deployment description:** An empty text input field.

At the bottom right of the dialog, there are two buttons: 'Cancel' and 'Deploy'.

STEP 12: CREATE THE APIS TO EXPOSE THE LAMBDA FUNCTIONS

Select the stages tab and note the invoke url.

The screenshot shows the AWS Management Console interface for the Amazon API Gateway. The breadcrumb navigation indicates the path: Amazon API Gateway > APIs > serverless-webapp-api (pywxuroz9c) > Stages > production. The left-hand navigation pane shows the 'Stages' tab selected under the 'production' API. The main content area is the 'production Stage Editor'. At the top, the 'Invoke URL' is displayed as `https://pywxuroz9c.execute-api.eu-central-1.amazonaws.com/production`, which is highlighted with a blue box and an orange arrow. Below this, there are tabs for 'Settings', 'Logs', 'Stage Variables', 'SDK Generation', and 'Export'. The 'Settings' tab is active, showing 'Cache Settings' with an 'Enable API cache' checkbox and 'Default Method Throttling' settings, including an 'Enable throttling' checkbox.

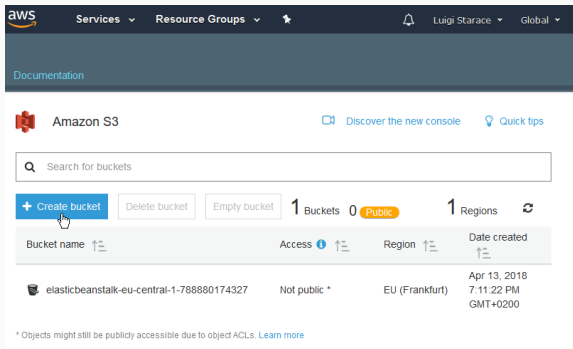
STEP 12: CREATE THE APIS TO EXPOSE THE LAMBDA FUNCTIONS

Change the CommentsAPI class accordingly in `src/API/CommentsAPI.js`.

```
class CommentsAPI {  
  
    constructor(){  
        this.endpoint = '<YOUR_INVOKE_URL_HERE>';  
    }  
  
    // ...  
  
}
```

STEP 13: HOST THE STATIC FILES WITH S3

Visit the S3 Dashboard and create a new bucket.



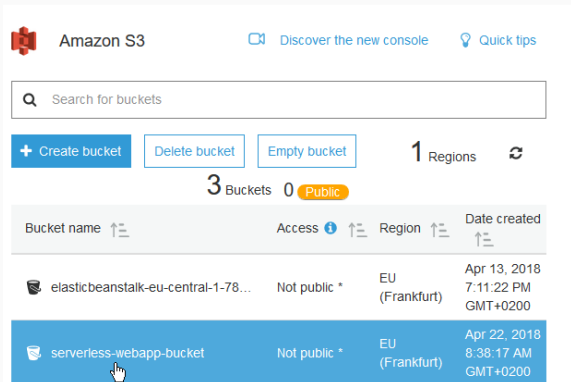
The screenshot shows the AWS Amazon S3 console dashboard. At the top, there is a navigation bar with the AWS logo, 'Services', 'Resource Groups', and a user profile for 'Luigi Starace'. Below this is a search bar for buckets. The dashboard displays a summary of resources: 1 Bucket, 0 Public buckets, and 1 Region. A table lists the existing bucket:

| Bucket name | Access | Region | Date created |
|--|--------------|----------------|--|
| elasticbeanstalk-eu-central-1-788880174327 | Not public * | EU (Frankfurt) | Apr 13, 2018 7:11:22 PM GMT+0200 |

* Objects might still be publicly accessible due to object ACLs. [Learn more](#)

STEP 13: HOST THE STATIC FILES WITH S3


Select your newly created S3 bucket.









The screenshot shows the Amazon S3 console interface. At the top, there is a search bar labeled "Search for buckets". Below the search bar are three buttons: "Create bucket" (highlighted in blue), "Delete bucket", and "Empty bucket". To the right of these buttons, it says "1 Regions" and "3 Buckets 0 Public". Below this is a table of buckets with columns for "Bucket name", "Access", "Region", and "Date created". The table contains two rows. The first row is for "elasticbeanstalk-eu-central-1-78..." with "Not public" access, "EU (Frankfurt)" region, and "Apr 13, 2018 7:11:22 PM GMT+0200" date. The second row is for "serverless-webapp-bucket" with "Not public" access, "EU (Frankfurt)" region, and "Apr 22, 2018 8:38:17 AM GMT+0200" date. A mouse cursor is pointing at the "serverless-webapp-bucket" row, which is highlighted in blue.

Amazon S3 [Discover the new console](#) [Quick tips](#)

Search for buckets

+ Create bucket Delete bucket Empty bucket 1 Regions 

3 Buckets 0 Public

| Bucket name  | Access  | Region  | Date created  |
|---|--|--|--|
|  elasticbeanstalk-eu-central-1-78... | Not public * | EU (Frankfurt) | Apr 13, 2018 7:11:22 PM GMT+0200 |
|  serverless-webapp-bucket | Not public * | EU (Frankfurt) | Apr 22, 2018 8:38:17 AM GMT+0200 |

STEP 13: HOST THE STATIC FILES WITH S3

Under the properties tab, select the Static website hosting card.

The screenshot shows the AWS Management Console interface for an Amazon S3 bucket named 'serverless-webapp-bucket'. The 'Properties' tab is selected, and the 'Static website hosting' card is highlighted with a mouse cursor. The console header includes the AWS logo, navigation menus for 'Services' and 'Resource Groups', a user profile for 'Luigi Starace', and a notification bell. The breadcrumb trail shows 'Amazon S3 > serverless-webapp-bucket'. The 'Properties' tab is active, and the 'Static website hosting' card is selected. The card contains the following text: 'Static website hosting', 'Host a static website, which does not require server-side technologies.', and a 'Learn more' link. Below the text is a 'Disabled' toggle switch. Other cards visible include 'Versioning', 'Server access logging', and 'Object-level logging', all of which are also disabled.

aws Services Resource Groups Luigi Starace

Amazon S3 > serverless-webapp-bucket

Overview Properties Permissions Managed

Versioning
Keep multiple versions of an object in the same bucket.
[Learn more](#)
Disabled

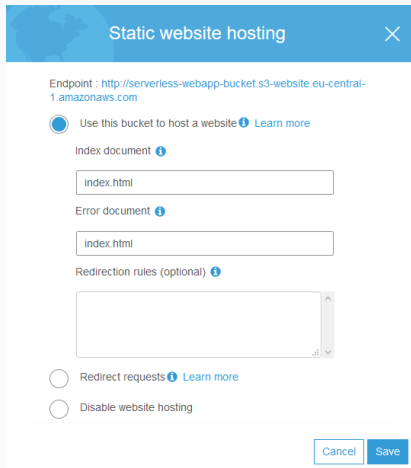
Server access logging
Set up access log records that provide details about access requests.
[Learn more](#)
Disabled

Static website hosting
Host a static website, which does not require server-side technologies.
[Learn more](#)
Disabled

Object-level logging
Record object-level API activity using the CloudTrail data events feature (additional cost).
[Learn more](#)
Disabled

STEP 13: HOST THE STATIC FILES WITH S3

Fill the form as shown in the picture. Note the Endpoint, as it will be the URL of the website!



The screenshot shows the 'Static website hosting' configuration form. At the top, there is a blue header with a globe icon, the text 'Static website hosting', and a close button (X). Below the header, the 'Endpoint' is displayed as 'http://serverless-webapp-bucket.s3-website.eu-central-1.amazonaws.com'. There are three radio button options: 'Use this bucket to host a website' (which is selected), 'Redirect requests', and 'Disable website hosting'. Each option has a blue information icon and a 'Learn more' link. Under the selected option, there are three input fields: 'Index document' (containing 'index.html'), 'Error document' (containing 'index.html'), and 'Redirection rules (optional)' (an empty text area). At the bottom right, there are 'Cancel' and 'Save' buttons.

Static website hosting

Endpoint : `http://serverless-webapp-bucket.s3-website.eu-central-1.amazonaws.com`

Use this bucket to host a website [Learn more](#)

Index document [?](#)

Error document [?](#)

Redirection rules (optional) [?](#)

Redirect requests [Learn more](#)

Disable website hosting

STEP 13: HOST THE STATIC FILES WITH S3

Now we'll show how to upload the static website via AWS CLI. This operation can also be performed via the web interface of the bucket.

STEP 13: HOST THE STATIC FILES WITH S3

Install AWS CLI

```
D:\serverless-webapp> pip install awscli --upgrade
```

Then configure it

```
D:\serverless-webapp> aws configure
AWS Access Key ID [None]: <YOUR_ACCESS_KEY_ID>
AWS Secret Access Key [None]: <YOUR_SECRET_ACCESS_KEY>
Default region name [None]: eu-central-1
Default output format [None]: json
```

STEP 13: HOST THE STATIC FILES WITH S3

Build the website

```
D:\serverless-webapp> npm run build-css  
D:\serverless-webapp> npm run build
```

Then upload the files with

```
D:\serverless-webapp> aws s3 sync ./build s3://  
serverless-webapp-bucket --acl public-read
```

STEP 13: HOST THE STATIC FILES WITH S3

After the upload is done, your bucket should look like this.

The screenshot shows the AWS Management Console interface for an Amazon S3 bucket. The breadcrumb navigation indicates the path: Amazon S3 > serverless-webapp-bucket. The console has tabs for Overview, Properties, Permissions, and Management. A search bar is present with the placeholder text "Type a prefix and press Enter to search. Press ESC to clear." Below the search bar are buttons for "Upload", "Create folder", and "More". The region is set to "EU (Frankfurt)". A table displays the bucket's contents, showing a folder named "static" and five files: "asset-manifest.json", "favicon.ico", "index.html", "manifest.json", and "service-worker.js". Each file entry includes a checkbox, an icon, the file name, the last modified date and time (all on Apr 22, 2018 at 8:50:08 AM GMT+0200), the size, and the storage class (all Standard). The table is currently displaying items 1 to 6.

| <input type="checkbox"/> | Name | Last modified | Size | Storage class |
|--------------------------|---------------------|----------------------------------|---------|---------------|
| <input type="checkbox"/> | static | -- | -- | -- |
| <input type="checkbox"/> | asset-manifest.json | Apr 22, 2018 8:50:08 AM GMT+0200 | 263.0 B | Standard |
| <input type="checkbox"/> | favicon.ico | Apr 22, 2018 8:50:08 AM GMT+0200 | 3.8 KB | Standard |
| <input type="checkbox"/> | index.html | Apr 22, 2018 8:50:08 AM GMT+0200 | 742.0 B | Standard |
| <input type="checkbox"/> | manifest.json | Apr 22, 2018 8:50:08 AM GMT+0200 | 332.0 B | Standard |
| <input type="checkbox"/> | service-worker.js | Apr 22, 2018 8:50:08 AM GMT+0200 | 3.2 KB | Standard |

STEP 13: HOST THE STATIC FILES WITH S3

Visit the Static website hosting card again under the properties tab, then click on the endpoint URL.

Static website hosting ✕

Endpoint : <http://serverless-webapp-bucket-s3-website.eu-central-1.amazonaws.com>

Use this bucket to host a website [Learn more](#)

Index document [?](#)

Error document [?](#)

Redirection rules (optional) [?](#)

Redirect requests [Learn more](#)

Disable website hosting

Cancel Save

STEP 13: HOST THE STATIC FILES WITH S3

You should see a very nice single-page serverless web application!

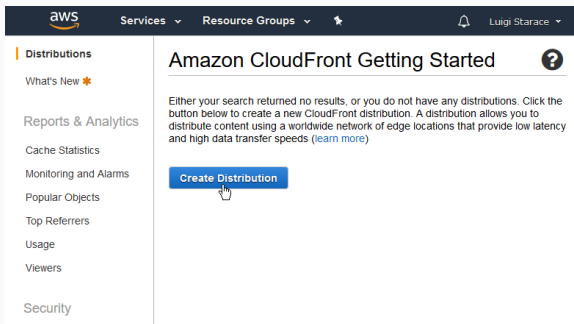


But wait, there's more!

Such a nice web application wouldn't be complete without a global CDN to speed up load times. So we'll now set up Amazon CloudFront to distribute the static files all over the globe.

STEP 14: OPTIMIZE LATENCY WITH CLOUDFRONT

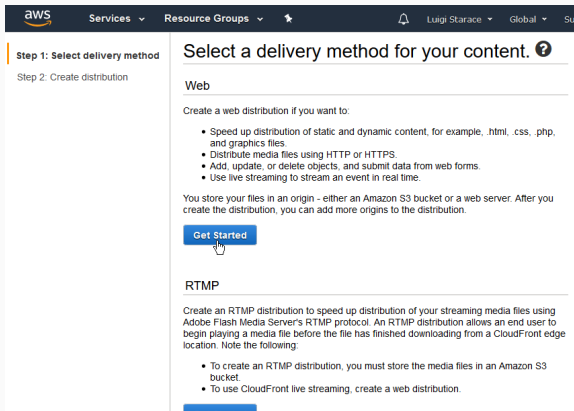
Visit the CloudFront Dashboard and create a new distribution.



The screenshot shows the AWS CloudFront console interface. At the top, the AWS logo is on the left, and navigation options for 'Services', 'Resource Groups', and a user profile 'Luigi Starace' are on the right. A left-hand navigation menu lists various sections: 'Distributions' (highlighted with an orange bar), 'What's New', 'Reports & Analytics', 'Cache Statistics', 'Monitoring and Alarms', 'Popular Objects', 'Top Referrers', 'Usage', 'Viewers', and 'Security'. The main content area is titled 'Amazon CloudFront Getting Started' with a help icon. Below the title, a message states: 'Either your search returned no results, or you do not have any distributions. Click the button below to create a new CloudFront distribution. A distribution allows you to distribute content using a worldwide network of edge locations that provide low latency and high data transfer speeds (learn more)'. A prominent blue button labeled 'Create Distribution' is centered in the main area, with a mouse cursor hovering over it.

STEP 14: OPTIMIZE LATENCY WITH CLOUDFRONT

Select web delivery method.



The screenshot shows the AWS CloudFront console interface. At the top, the navigation bar includes the AWS logo, 'Services', 'Resource Groups', and a user profile for 'Luigi Starace'. The main content area is titled 'Select a delivery method for your content.' and features a sidebar with 'Step 1: Select delivery method' (highlighted) and 'Step 2: Create distribution'. The 'Web' section is selected, showing instructions to create a web distribution and a list of benefits: speeding up static/dynamic content, distributing media files, adding/deleting objects, and using live streaming. A 'Get Started' button is highlighted with a mouse cursor. The 'RTMP' section is also visible below, with its own 'Get Started' button partially shown.

aws Services Resource Groups Luigi Starace Global

Step 1: Select delivery method

Step 2: Create distribution

Select a delivery method for your content. ?

Web

Create a web distribution if you want to:

- Speed up distribution of static and dynamic content, for example, .html, .css, .php, and graphics files.
- Distribute media files using HTTP or HTTPS.
- Add, update, or delete objects, and submit data from web forms.
- Use live streaming to stream an event in real time.

You store your files in an origin - either an Amazon S3 bucket or a web server. After you create the distribution, you can add more origins to the distribution.

[Get Started](#)

RTMP

Create an RTMP distribution to speed up distribution of your streaming media files using Adobe Flash Media Server's RTMP protocol. An RTMP distribution allows an end user to begin playing a media file before the file has finished downloading from a CloudFront edge location. Note the following:

- To create an RTMP distribution, you must store the media files in an Amazon S3 bucket.
- To use CloudFront live streaming, create a web distribution.

[Get Started](#)

STEP 14: OPTIMIZE LATENCY WITH CLOUDFRONT

Select your website bucket as the origin.

Create Distribution ?

Origin Settings

| | |
|------------------------|--|
| Origin Domain Name | serverless-webapp-bucket.s3.amazonaws.com |
| Origin Path | — Amazon S3 Buckets — serverless-webapp-bucket.s3.amazonaws.com |
| Origin ID | S3-serverless-webapp-bucket |
| Restrict Bucket Access | <input type="radio"/> Yes <input checked="" type="radio"/> No |
| Origin Custom Headers | Header Name <input type="text"/> |

STEP 14: OPTIMIZE LATENCY WITH CLOUDFRONT

Select index.html as the default root object.

Supported HTTP Versions HTTP/2, HTTP/1.1, HTTP/1.0 HTTP/1.1, HTTP/1.0 ⓘ

Default Root Object ⓘ

Logging On Off ⓘ

Bucket for Logs ⓘ

Log Prefix ⓘ

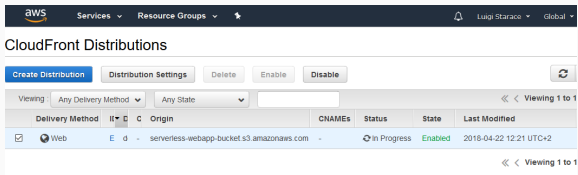
Cookie Logging On Off ⓘ

Enable IPv6 ⓘ
[Learn more](#)

Comment ⓘ

STEP 14: OPTIMIZE LATENCY WITH CLOUDFRONT

It takes a few minutes to setup the distribution. When it's done the status will change to Deployed.

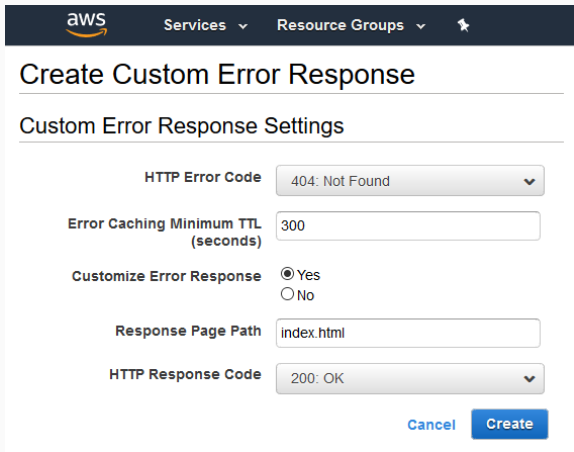


The screenshot shows the AWS CloudFront Distributions console. At the top, there are navigation tabs: "Create Distribution" (highlighted in blue), "Distribution Settings", "Delete", "Enable", and "Disable". Below the tabs, there are filters for "Viewing" (Any Delivery Method), "Any State", and a search box. The main content is a table with the following columns: "Delivery Method", "ID", "Origin", "CNAMEs", "Status", "State", and "Last Modified".

| Delivery Method | ID | Origin | CNAMEs | Status | State | Last Modified |
|-----------------|-----|---|--------|-------------|---------|------------------------|
| Web | E d | serverless-webapp-bucket.s3.amazonaws.com | - | In Progress | Enabled | 2018-04-22 12:21 UTC+2 |

STEP 14: OPTIMIZE LATENCY WITH CLOUDFRONT

In the distribution detail page add a custom error response as shown in the picture below to make sure 404 errors are handled by the application.



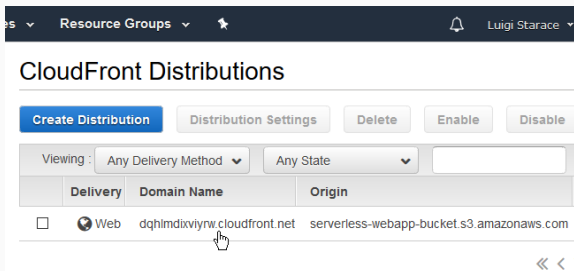
The screenshot shows the AWS CloudFront console interface for creating a custom error response. At the top, the AWS logo is on the left, and 'Services' and 'Resource Groups' are on the right. The main heading is 'Create Custom Error Response'. Below it is the section 'Custom Error Response Settings'. The settings are as follows:

- HTTP Error Code:** A dropdown menu set to '404: Not Found'.
- Error Caching Minimum TTL (seconds):** A text input field containing '300'.
- Customize Error Response:** Radio buttons for 'Yes' (selected) and 'No'.
- Response Page Path:** A text input field containing 'index.html'.
- HTTP Response Code:** A dropdown menu set to '200: OK'.

At the bottom right, there are two buttons: 'Cancel' and 'Create'.

STEP 14: OPTIMIZE LATENCY WITH CLOUDFRONT

Now you can visit the application from the cloudfront URL



The screenshot shows the AWS CloudFront console interface. At the top, there's a navigation bar with "Resource Groups" and a user profile "Luigi Starace". The main heading is "CloudFront Distributions". Below the heading are several action buttons: "Create Distribution" (highlighted in blue), "Distribution Settings", "Delete", "Enable", and "Disable".

Below the buttons, there are filters: "Viewing:" with "Any Delivery Method" and "Any State" dropdowns, and an empty search input field.

The main content is a table with the following columns: "Delivery", "Domain Name", and "Origin".

| | Delivery | Domain Name | Origin |
|--------------------------|----------|------------------------------|---|
| <input type="checkbox"/> | Web | dqhlmdixviryw.cloudfront.net | serverless-webapp-bucket.s3.amazonaws.com |

At the bottom right of the table, there are navigation arrows: "« <". A mouse cursor is pointing at the domain name "dqhlmdixviryw.cloudfront.net".

TAKE HOME MESSAGES

- Cloud computing and service models

- Cloud computing and service models
- AWS

- Cloud computing and service models
- AWS
- Deploy a “classic” web application on AWS

- Cloud computing and service models
- AWS
- Deploy a “classic” web application on AWS
- FaaS and serverless computing

- Cloud computing and service models
- AWS
- Deploy a “classic” web application on AWS
- FaaS and serverless computing
- Build and deploy a serverless one-page web application on AWS

SECURITY RECOMMENDATIONS

- Be **very** careful not to expose your IAM credentials;

SECURITY RECOMMENDATIONS

- Be **very** careful not to expose your IAM credentials;
- Enforce the least privilege principle: each user should only be able to access the minimum resources necessary to fulfill its purpose.

SECURITY RECOMMENDATIONS

- The very second you expose your credentials to the public, some bot may use them to spin up large numbers of EC2 instances. If that happens, the billing might be a scary surprise!

⏪ Back to the tutorial

IAM best practices  web

Git Secrets - Github repository  web

SECURITY RECOMMENDATIONS

- The very second you expose your credentials to the public, some bot may use them to spin up large numbers of EC2 instances. If that happens, the billing might be a scary surprise!
- Tools like the AWS-developed `git-secrets` help avoiding the exposure of IAM credentials

⏪ Back to the tutorial

IAM best practices  web

Git Secrets - Github repository  web

- [Jan16] Badri Janakiraman. *Serverless*. June 20, 2016. URL: <https://martinfowler.com/bliki/Serverless.html> (visited on 05/21/2018).
- [LF14] James Lewis and Martin Fowler. *Microservices: a definition of this new architectural term*. Mar. 25, 2014. URL: <https://martinfowler.com/articles/microservices.html> (visited on 05/21/2018).
- [Rob16] Mike Roberts. *Serverless Architectures*. Apr. 6, 2016. URL: <https://martinfowler.com/articles/serverless.html> (visited on 05/21/2018).

- [Rus16] Mark Russinovich. *Microservices: An application revolution powered by the cloud*. Mar. 17, 2016. URL: <https://azure.microsoft.com/it-it/blog/microservices-an-application-revolution-powered-by-the-cloud/> (visited on 05/21/2018).
- [Ser] Serverless inc. *Serverless guide*. URL: <https://github.com/serverless/guide> (visited on 05/21/2018).

REFERENCES I

- [Amaa] Amazon Web Services. *What is cloud computing?*. URL: <https://aws.amazon.com/what-is-cloud-computing/> (visited on 03/30/2018).
- [Amab] Inc. Amazon Web Services. *Set up a Continuous Deployment Pipeline using AWS CodePipeline*. URL: <https://aws.amazon.com/it/getting-started/tutorials/continuous-deployment-pipeline/> (visited on 06/10/2018).

REFERENCES II

- [Amac] Inc. Amazon Web Services. *Tutorial: Create a Four-Stage Pipeline*. URL: <https://docs.aws.amazon.com/codepipeline/latest/userguide/tutorials-four-stage-pipeline.html> (visited on 06/10/2018).
- [Ama17] Inc. Amazon Web Services. *Practicing Continuous Integration and Continuous Delivery on AWS*. Tech. rep. June 2017. URL: <https://d1.awsstatic.com/whitepapers/DevOps/practicing-continuous-integration-continuous-delivery-on-AWS.pdf> (visited on 06/01/2018).

REFERENCES III

- [AWS] AWS. *AWS Step Functions*. URL: https://aws.amazon.com/step-functions/?nc1=f_ls (visited on 05/01/2018).
- [Gar17] Gartner. *Gartner Forecasts Worldwide Public Cloud Services Revenue to Reach \$260 Billion in 2017*. Oct. 12, 2017. URL: <https://www.gartner.com/newsroom/id/3815165> (visited on 03/30/2018).

- [Syn18] Synergy Research Group. *Cloud Growth Rate Increases; Amazon, Microsoft & Google all Gain Market Share*. Feb. 2, 2018. URL: <https://www.srgresearch.com/articles/cloud-growth-rate-increases-amazon-microsoft-google-all-gain-market-share> (visited on 03/30/2018).